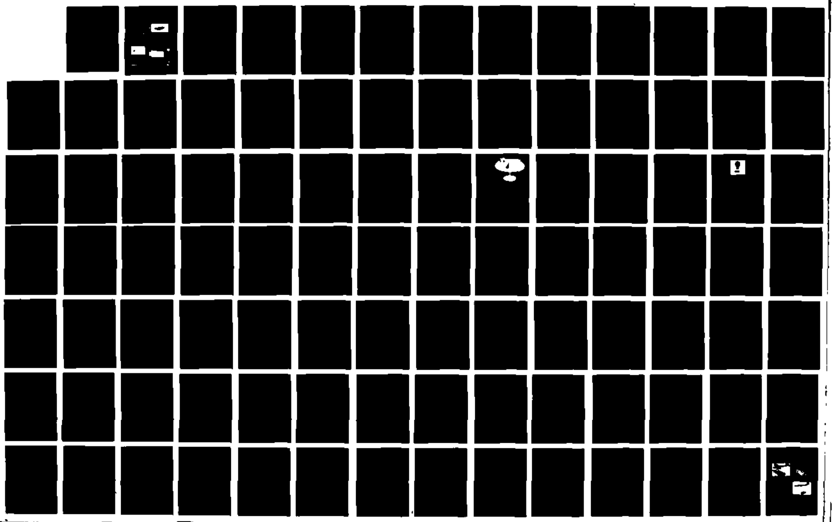
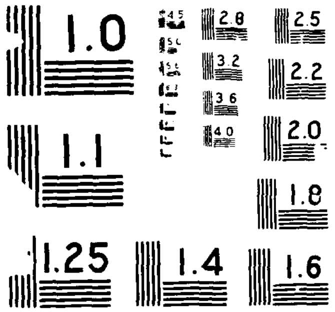
AD-A212 981 PROCEEDINGS OF IMAGE UNDERSTANDING WORKSHOP HELD IN 1/13
PALO ALTO CALIFORNIA O.. (U) SCIENCE APPLICATIONS
INTERNATIONAL CORP MCLEAN VA L S BAUMAN MAY 89
UNCLASSIFIED N00014-86-C-0700 F/G 17/11 NL

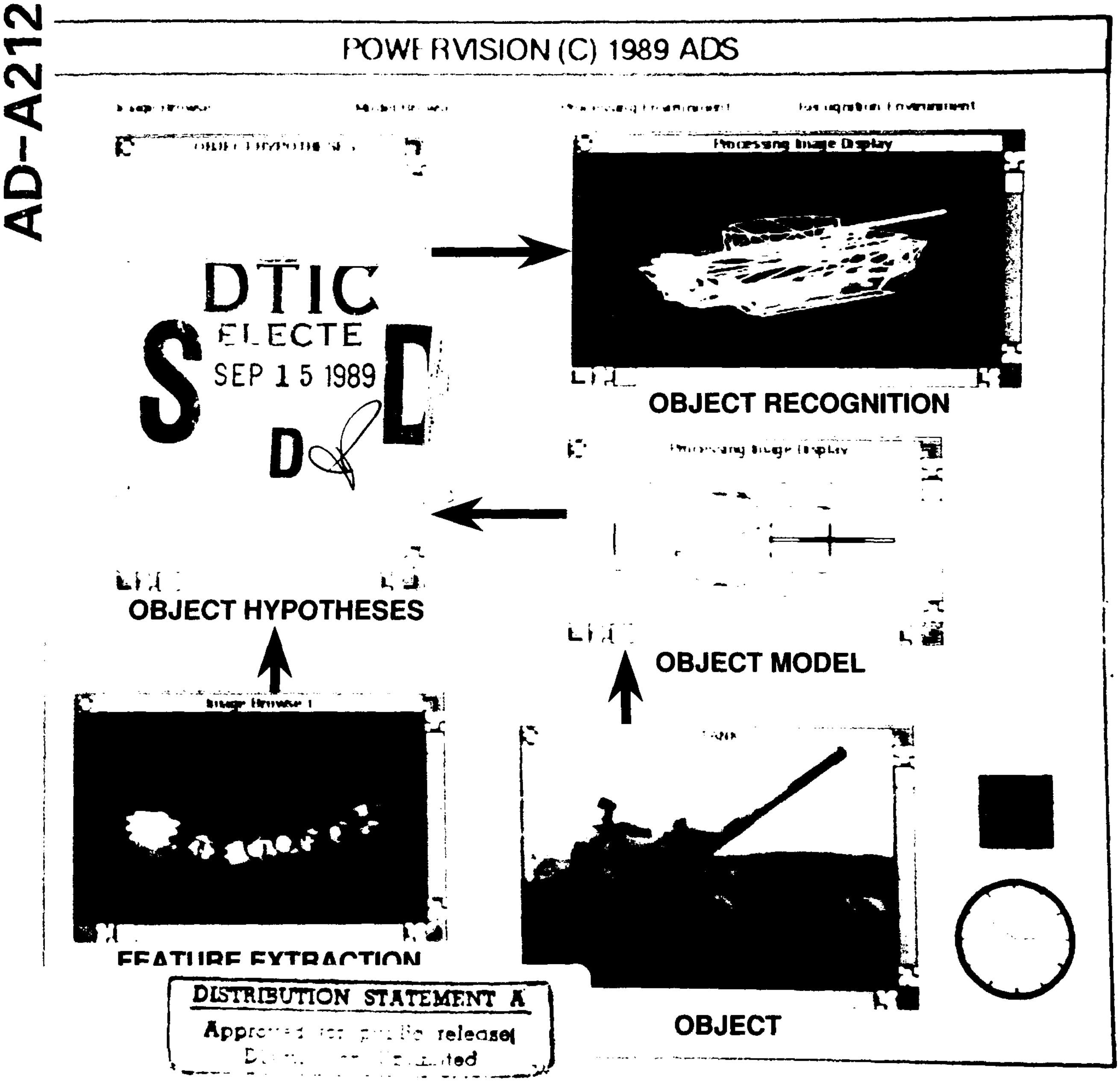



1.0
4.5
5.0
5.6
2.8
2.5
3.2
2.2
3.6
4.0
2.0
1.1
1.8
1.25
1.4
1.6

(2)

PROCEEDINGS:

Image Understanding Workshop



Sponsored by:
Defense Advanced Research Projects Agency
Information Science and Technology Office

DARPA

SC

May 1989

AD A212 781

SAIC

Science Applications International Corporation
An Employee Owned Company

September 12, 1989

Acquisition Section
Defense Logistics Agency
Defense Technical Information Center
Cameron Station
Alexandria, VA 22304-6145

Dear Sir:

Two copies of the Proceedings for the May 1989 Image Understanding workshop are enclosed for your use. Although these proceeedings were published by Morgan Kaufman Publishers, no copyright is involved and the publisher is aware of DARPA's intention to provide the report to the DTIC as unclassified, approved for unlimited distribution.

A DD Form 1473 for insertion in the report and a DTIC Form 50 for notification of the AD number are enclosed.

Yours truly,

SCIENCE APPLICATIONS
INTERNATIONAL CORPORATION

file - FDAC

Lee S. Baumann

Enclosures

UNCLASSIFIED

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

REPORT DOCUMENTATION PAGE

READ INSTRUCTIONS BEFORE COMPLETING FORM

1. REPORT NUMBER

2. GOVT ACCESSION NO.

3. RECIPIENT'S CATALOG NUMBER

4. TITLE (and Subtitle)

Proceedings:
Image Understanding Workshop
May 1989

5. TYPE OF REPORT & PERIOD COVERED

ANNUAL TECHNICAL
April 1988-May 1989

6. PERFORMING ORG. REPORT NUMBER

7. AUTHOR(s)

Lee S. Baumann (Editor)

8. CONTRACT OR GRANT NUMBER(s)

N00014-86-C-0700

9. PERFORMING ORGANIZATION NAME AND ADDRESS

SCIENCE APPLICATIONS INTERNATIONAL CORPORATION
1710 Goodridge Drive
McLean, VA 22102

10. PROGRAM ELEMENT, PROJECT, TASK AREA & WORK UNIT NUMBERS

ARPA Order 5605

11. CONTROLLING OFFICE NAME AND ADDRESS

Defense Advanced Research Projects Agency
1400 Wilson Blvd.
Arlington, VA 22209

12. REPORT DATE

May 1989

13. NUMBER OF PAGES

1151

14. MONITORING AGENCY NAME & ADDRESS(if different from Controlling Office)

15. SECURITY CLASS. (of this report)

UNCLASSIFIED

15a. DECLASSIFICATION/DOWNGRADING SCHEDULE

16. DISTRIBUTION STATEMENT (of this Report)

APPROVED FOR PUBLIC RELEASE, DISTRIBUTION UNLIMITED

17. DISTRIBUTION STATEMENT (of the abstract entered in Block 20, if different from Report)

18. SUPPLEMENTARY NOTES

19. KEY WORDS (Continue on reverse side if necessary and identify by block number)

Digital Image Processing; Image Understanding; Scene Analysis; Edge Detection; Image Segmentation; CCD Arrays; CCD Processors

20. ABSTRACT (Continue on reverse side if necessary and identify by block number)

This document contains the annual progress reports and technical papers presented by the research activities in Image Understanding sponsored by the Information Science & Technology Office, Defense Advanced Research Projects Agency. The papers were presented at a workshop conducted on 23-26 May 1989, in Palo Alto, California. Also included are copies of Special Reports presented at the workshop and additional technical papers from the research activities which were not presented due to lack of time but are germane to this research field.

UNCLASSIFIED

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

Image Understanding Workshop

Proceedings of a Workshop
Held at
Palo Alto, California

May 23-26, 1989

Sponsored by:
Defense Advanced Research Projects Agency
Information Science and Technology Office

This document contains copies of reports prepared for the DARPA Image Understanding Workshop. Included are Principal Investigator reports and technical results from both the basic and strategic computing programs within DARPA/ISTO sponsored projects and certain technical reports from selected scientists from other organizations.

The views and conclusions contained in this document are those of the authors and should not be interpreted as necessarily representing the official policies, either expressed or implied, of the Defense Advanced Research Projects Agency or the United States Government.

Distributed by

Morgan Kaufmann Publishers, Inc.
2929 Campus Drive
San Mateo, California 94403

ISBN 1-55860-070-1

Printed in the United States of America

TABLE OF CONTENTS

SECTION II: SPECIAL TALKS & PRESENTED TECHNICAL PAPERS

Session 2 - Parallel Processing

Session 5 - Special Talks

Session 6 - Low Level Vision

PAGE

SECTION III: OTHER TECHNICAL REPORTS

Parallel Processing

Applications

PAGE

High Level Vision

Low Level Vision

AUTHOR INDEX

ACKNOWLEDGMENTS

The nineteenth computer Image Understanding Workshop was held in Palo Alto, California on May 23-26, 1989. This workshop, attended by more than two hundred research and government personnel, was sponsored by the Information Science and Technology Office of the Defense Advanced Research Projects Agency (DARPA).

The main theme of the 1989 workshop was announced by Lt. Col. Robert L. Simpson, Jr., the Program Manager for Machine Intelligence, as "High Level Vision." In addition to this main theme, papers were presented on parallel processing, low level vision, and vision applications. As usual, the main objectives of the workshop are to review the latest results in the image understanding research program, help keep the government research community aware of evolving technology, and most importantly, to exchange ideas, needs and trends in computer vision research. Mr. Robert Drazovich of Advanced Decision Systems coordinated the technical program for this workshop.

The papers appearing in this proceedings were prepared by the research personnel of the various institutions involved in computer vision research. The first section is comprised of an overview paper by the program manager and the research reviews presented by the principal investigators of the research institutions included in the DARPA program. The second section contains the technical papers presented by members of the technical staffs and guest speakers at the workshop. The final section is composed of submitted technical papers which were not presented due to lack of time but which are included in order to present as full a record as is possible of recent results in the computer vision field.

The figure appearing on the cover was prepared by Phil Kahn and Kathleen Wilson of Advanced Decision Systems (ADS). It represents a model-based vision paradigm in which image objects are interpreted by using models to predict image features and by forming hypotheses about the association between extracted and predicted objects. The processing paradigm is undertaken within an object-oriented, image understanding environment. ADS would like to thank Wright-Patterson Air Force Base for making the imagery available.

The artwork and layout designs were done by Mr. Tom Dickerson of the SAIC graphics staff. Significant assistance in handling the mailings, and in putting the proceedings together for publication was provided by Ms. Jayne Karim and Ms. Mary Bourassa of the Science Applications International Corporation staff.

SECTION I

OVERVIEW

AND

PRINCIPAL INVESTIGATOR

REPORTS

An Update On
STRATEGIC COMPUTING COMPUTER VISION

Robert L. Simpson, Jr., Ph.D.
Lieutenant Colonel, USAF
Program Manager for Machine Intelligence
Defense Advanced Research Projects Agency
Information Science and Technology Office
Arlington, VA 22209-2308

ABSTRACT

Since 1985, I have had the privilege of managing the DARPA Strategic Computing Computer Vision Technology Base, which is a sizable subset of the overall DARPA Image Understanding Program. After four years of very significant activity by this DARPA sponsored computer vision community conducting its research under the auspices of the Strategic Computing (SC) Program, I wish to take this opportunity to relate from a global perspective some of these activities and accomplishments of this critically important national research program. This paper is not going to present technical details; those can be gathered from the other papers in this volume. My purpose is to present some insight into the goals, structure, and relationships among the various projects sponsored under SC. While I believe you will agree that much has been accomplished, there is much more research and development needed before the ultimate goal of SC Computer Vision research is realized.

INTRODUCTION

The ultimate goal of SC computer vision research (SCVision) is to develop knowledge-based technology that will enable the construction of complete, robust, high performance image understanding systems to support a wide range of DoD applications. These applications include autonomous vehicle navigation, photointerpretation, smart weapons, manufacturing, and robotic manipulation. One of the featured areas of research in computer vision has been autonomous ground vehicles, since military vehicles frequently perform dangerous or labor-intensive tasks and in many cases, the capabilities required for a robotic vehicle appear to be within reach of a concerted research effort in robot vision, computer architectures, and machine planning technologies. The SCVision program is aimed at extending the state-of-the-art so that robotic systems will be able to perform such tasks. In addition, this domain is appropriate for challenging image understanding technology because it focuses on critical component vision technologies that are important to all of the relevant application areas in SC. For example, the navigation task for a ground vehicle requires the representation and understanding of complex dynamic scenes containing natural and man-made objects, and the development and use of new computer parallel architectures with new programming techniques to meet demanding real-time computational needs. These same issues are germane to photointerpretation, manufacturing, and smart weapons, where real-time computing issues are particularly critical in this last application. A significant part of the technological results are also being exploited in other DoD programs requiring similar model-based recognition and classification problems. Two such programs are the Advanced Digital Radar Imagery Exploitation System (ADRIES) and the Strategic Computing Object-Directed Reconnaissance Parallel-Processing Image Understanding System (SCORPIUS).

The SCVision program has focused on three types of demonstration scenarios for vision-guided robot vehicles: road following, cross-country navigation, and the unification of these first two scenarios into a single system to accomplish specific mission goals. Significant accomplishments have been demonstrated for the basic two scenarios, and current research is targeted at an integrated system. In road following scenarios, a vehicle drives down a stretch of paved or unpaved road. Research has dealt with the complexity of outdoor illumination and shadows, recognition of ground features such as road boundaries, and the avoidance of obstacles using 3-dimensional sensor data. Road following provides a well-defined task with clearly defined features of interest, so that basic capabilities for vision and mobility can be developed. Road following demonstrations have achieved the fastest vehicle speeds, the longest distances, and have had the greatest number of experiments performed to date.

In the successful demonstration of the cross-country navigation scenario, autonomous vehicles have been driven across natural terrain under guidance of 3-D sensors, particularly the laser range scanner. In these demonstrations, the goal point was not visible from the starting vehicle location. The research required unifying the technology for map-based reasoning with sensor-based planning techniques. The SCVision effort was aimed at classifying terrain as traversable or non-traversable using a 3-dimensional vehicle model, building a representation for observations of extended obstacles including slopes, ditches, and rock outcrops, and moving the vehicle safely and efficiently along an automatically planned path. The cross country experiments have successfully driven robots vehicles in complex terrain and difficult navigational environments.

Finally, an integrated scenario will require both on-road and off-road navigation capability as well as object recognition for landmarks and targets of interest. In an integrated system, the robot vehicle would be assigned a complex, multi-step mission and the vehicle itself would decide which navigation and perception modes were appropriate for each step of the task. An example mission that an integrated system might carry out is the "overwatch" reconnaissance mission in which the vehicle must navigate to a vantage point overlooking a strategically important location, position the vehicle to properly direct its sensors at the location, monitor targets appearing at the location, report on the target movement as specified by mission constraints, and finally return to its base of operations. Research for building integrated systems is needed for object and event recognition, dynamically interpreting perceived information to satisfy mission constraints, maintaining consistency with map-based knowledge, building flexible software system architectures, and incorporating high-speed computing hardware. In addition to developing individual component technologies, the SCVision program includes a new generation system (NGS) that will be such an integrated system.

To achieve the goals of all these scenarios, SCVision research addresses four critical areas in knowledge-based image understanding: visual modeling and recognition, dynamic scene motion analysis, vision-based obstacle avoidance and path planning, and parallel implementation issues. In addition, the NGS effort focuses on system integration problems. The goal of visual modeling is to develop effective and efficient techniques for describing, storing, and accessing the knowledge necessary for natural and man-made object interpretation. Natural objects (such as rocks, bushes, and ravines) and cultural objects (such as buildings and fences) must be modeled in such a way that their characteristics can be provided to effective recognition modules that locate occurrences of them in sensed data. Dynamic scene motion analysis interprets a sequence of images to determine the shape and location of objects and to detect and track moving objects. Vision-based obstacle avoidance and path planning research explores techniques to plan and follow routes, locate and follow roads, and detect and avoid obstacles on the basis of visual input. The research in parallel vision algorithms aims at developing vision programming environments for parallel computer architectures and implementing vision algorithms on them, so that the sensory analysis task is performed in real time. The system integration work is needed to develop system architectures that coordinate the activities of multiple sensor and reasoning systems to perform complex tasks. In addition, the NGS is being developed as a focus for integrating and testing research results. The purpose of the NGS research is to ensure that complex system issues are being addressed, to facilitate appropriate demonstrations and evaluations, and to achieve accelerated transfer of technology to applications.

SCVision, whose emphasis is on knowledge-based approaches to machine perception, is aimed at application tasks that cannot be successfully addressed with simple image matching and statistical analysis. For this reason, SCVision research is explicitly avoiding investigations that are based on statistical pattern matching, which has been the subject of extensive research over the past thirty years. In addition, SCVision is explicitly avoiding research whose predominant goal is the modeling or simulation of biological systems. SCVision also emphasizes the systems aspect of vision -- integrating vision with other AI and control modules in a real-time system, and investigating state-of-the-art hardware architectures and software systems to make practical use of new hardware in integrated systems.

ACCOMPLISHMENTS

The SCVision program has built on results from the first phase of research to achieve accomplishments in each area of the major technical problem areas. The accomplishments are described below

NEW GENERATION SYSTEM

An initial implementation and demonstration of the NGS has been completed at CMU. It is based on a blackboard architecture, with specialized data representations and access methods to support three-dimensional spatial reasoning. The NGS runs on the NAVLAB, a commercial van converted into a computer-controlled robot vehicle. The NAVLAB incorporates a multi-processor computing environment consisting of general-purpose computers (SUN3s) as well as a WARP parallel machine. The application software includes a number of vision modules capable of detecting and following roads and detecting and avoiding obstacles, using algorithms developed in the SCVision program. Perception modules handle sun, shade, and mixed illumination; modules using a laser rangefinder have even driven the NAVLAB at night. Using the WARP, the NAVLAB has been driven at 1 meter per second. The NGS allows road-following and obstacle-avoidance to be combined into a single system. Research during the past year integrated capabilities for cross-country travel, and for building maps and retraversing terrain with less perception and higher speeds. The new CORE system will integrate perception, planning, and path execution into a single framework. Data collected by the NAVLAB has been distributed to other SCVision sites, and several components of the NGS have been transferred to the ALV project at Martin Marietta in Denver and have been used to drive the ALV.

VISUAL MODELING AND RECOGNITION

Substantial progress has been made within the SCVision community for the basic techniques for visual modeling and recognition. SRI has completed an initial implementation of the Core Knowledge System (CKS) for vision systems to allow cooperative interactions among sensors, interpreters, controllers and user interfaces. The CKS, together with new techniques developed at SRI for recovering scene geometry and object identity from a sequence of images, provides a basis for understanding both the semantics and geometry of the environment being traversed by a moving robotic device. Using the superquadric model, UPenn has developed a parameter fitting algorithm to extract surface descriptions from clusters of 3D points from the laser range scanner. Surface parameters include position, orientation, scale, planar or quadratic surface measures, and two deformation parameters (tapering and bending along the major axis). Stanford has developed general methods for spectral modeling of objects based on analysis of linear projections of multispectral, optical images. ADS also developed an architecture for object modeling and recognition for autonomous land vehicles. Models of objects such as terrain features, horizon features, poles, trees and bushes have been developed at both ADS and Stanford. GE has demonstrated model-directed object recognition using vertex-pairs as matching features between model and scene. The algorithm was implemented on the Connection Machine and demonstrated for airplanes and automobiles. An extensive set of algorithms have been developed for recognizing roads and intersections by CMU and Maryland. These algorithms are based on modeling roads in terms of their spectral and geometric properties, and use both optical and range information. 3D range data analysis techniques have been advanced at CMU, Hughes, and SRI for recognizing terrain features and man-made objects. The University of Pennsylvania has developed techniques to use superquadrics for representing and recognizing landmarks and objects. Hughes has developed techniques to segment color imagery for the recognition of natural terrain objects.

DYNAMIC SCENE AND MOTION ANALYSIS

The University of Southern California has implemented techniques for robustly estimating 3D motion parameters and computing 3D motion trajectories of moving objects given features extracted from 2D image sequences. The theory has been developed for image sequences involving acceleration and deceleration. Relative structure and depth information is a by product of motion parameter estimation. To support this work, USC has developed line- and region-based feature extraction and matching techniques for detecting moving objects in motion sequences.

Honeywell has developed qualitative reasoning and "3-1/2-D" modeling techniques for detecting and tracking moving targets from a mobile platform in simple curved road scenes. The concept of fuzzy focus of expansion, which allows a very accurate determination of the instantaneous direction of a moving vehicle and camera rotations along the two axes (pan and tilt only), has been demonstrated. Honeywell has also demonstrated the "dynamic model matching" concept for landmark recognition, where the model generation and matching process dynamically changes as a function of range to the landmark and perspective as viewed by a mobile platform. In addition, they have performed initial experiments in digital map integrated target tracking.

The University of Massachusetts has collected a set of motion sequences from the Martin Marietta ALV which contain ground truth information from the vehicle's land navigation system, as well as a cartographic survey of the environment; these sequences will be made available to the SCVision community to allow scientific evaluation of motion algorithms. UMass also has developed a promising algorithm for depth computation from image sequences using optic flow fields to track straight lines.

Hughes has implemented a technique developed by MIT to recover vehicle motion (with 6 degrees of freedom) between two locations using information from laser range scans. This has allowed Hughes to construct topographical terrain maps from sensed data collected during cross country experiments on the ALV

OBSTACLE DETECTION AND AVOIDANCE

The SCVision community has developed and demonstrated a series of algorithms for detecting obstacles. The first algorithms developed for obstacle detection were fast, but could only be used for flat roads -- crowned or banked roads, or cross-country terrain, could not be handled by such simple approaches. The SCVision community has continued to develop and demonstrate a series of algorithms for detecting obstacles. Algorithms for complex conditions, crowned or banked roads, or cross-country terrain, have been developed at Maryland, using derivatives of range; at Hughes, by interpreting an elevation map with a three dimensional vehicle model; and at CMU, fitting multiple planes with descriptors for surface roughness and slope. At the University of Pennsylvania, research has emphasized active vision techniques and a control schema of peripheral and foveal imaging has been completed. The CMU algorithms have successfully driven vehicles through obstacles, both on and off road. The Hughes system was used in several cross country experiments with the ALV in December 1987, successfully avoiding obstacles such as trees, rocks, steep slopes, and gullies with paths of approximately 700 meters. This demonstration represented the integration of map and sensor-based operation of a robotic vehicle in natural terrain using tightly coupled vision and planning systems. It is significant that with this demonstration the DARPA cross country technology milestone was achieved approximately one year ahead of schedule. UPenn research has received considerable recognition for their wavelet decomposition of visual signals and the use of superquadrics for shape descriptions is being evaluated by the post office for use in separating irregular parcels on the assembly line.

PARALLEL COMPUTING ENVIRONMENTS FOR VISION

Many software tools and algorithms have been implemented on the different parallel architectures, Warp, The Connection Machine, and Butterfly. A significant body of experience has been gained in implementing vision algorithms on parallel computers. Now that the hardware has been available for some time, research has focussed on software programming environments for parallel vision. Carnegie-Mellon has implemented the Apply language, which allows the same low-level vision program to be run efficiently on a variety of parallel architectures. Apply currently generates efficient code for Warp and Sun, as well as other versions of Warp and other parallel architectures. Development of Apply has gone in two directions: mapping Apply onto new parallel architectures, and extending Apply's functionality on existing architectures, particularly Warp. MIT had concentrated in using the Connection Machine to explore parallel models of computation and for developing a vision system for unconstrained environments -- the Vision Machine. MIT demonstrated for the first time the whole Vision Machine system working from images to recognition through integration of other cues. Some of those algorithms run in close to real time: edge detection, motion computation, stereo, surface interpolation, fusion of several sources of visual information based on Markov Random Field models, and model-based recognition. Maryland has designed a pyramid image processing system with both fat and redundant pyramids and implemented many algorithms on the Connection Machine. Rochester has demonstrated SIMD-like programs on the BBN Butterfly Parallel Processor that show linear parallel speedup. Rochester has implemented key routines from the IFF image processing library on the Butterfly using the Uniform System, distributed by BBN and improved locally. Columbia has also built an image processing environment that optimally embeds tree and pyramid architectures in the Connection Machine. Columbia has developed and demonstrated new algorithms for surface shape recovery from textural cues, and a system for optimally recovering surface properties from sparse stereo or range data.

TECHNOLOGY TRANSFER

Several of the important components developed under SCVision have been transferred to applications. For the ALV project, modules from CMU and Maryland have been incorporated into the Martin Marietta system, and complete systems from Hughes and CMU have driven the ALV on roads and cross country. In particular, the NGS architecture (hardware and software) is now been duplicated on the ALV, facilitating transfer of complete modules. NGS technology is also being used in the design of a Mars Rover and in studies of underwater autonomous vehicles. Several sites are using SCVision parallel vision software. Connection Machine, Warp, and Butterfly software have been exported to DoD projects such as ADRIES and SCORPIUS, and as far as the European Community's ECRC

PROJECT SUMMARIES

NEW GENERATION VISION SYSTEM (CMU)

The New Generation Vision System, being developed by CMU, is an integrated vision system based on the blackboard architecture and containing vision modules developed at CMU based on the work of CMU and other SCVision contractors. Current capabilities of the NGS include road following, simple obstacle avoidance, and cross-country driving. The blackboard provides communications and coordination between sensing and reasoning modules. The testbed for this system the CMU NAVLAB, a commercial truck converted into a robot vehicle. The NAVLAB contains several general-purpose computer workstations on board as well as a WARP parallel architecture computer; sensors include color TV cameras and an ERIM laser rangefinder. Researchers on the NAVLAB can monitor the execution of the system as it is running, which has been invaluable in speeding up research progress and acquiring data for distribution to all the SCVision sites. The NGS software runs on standard UNIX, which facilitates program development and testing, and the WARP software includes fully integrated programming tools and a vision library of over 200 routines. The system evolves over time as new modules are integrated and as the hardware and software environments evolve. In particular, to consolidate the research progress of the program to date, the ACORN Core Navigation System is being developed to integrate the most successful and reliable versions of range data interpretation, cross-country path planning (based on work done at Hughes), and vehicle control. ACORN will provide a self-contained software platform for the NGS so that further perception research can be undertaken without the need to constantly redefine the basic vehicle control and navigation facilities. Major achievements are: the CODGER blackboard for mobile robot development; the Driving Pipeline architecture for continuous perception and motion control; fast vision using Warp and enhanced W2 language; following a road at 1 meter / second; road following using active sensing for day / night capability; mapping 3-D features and reusing the map to guide later runs; building detailed 3-D maps for cross-country traverse and planning trajectories based on the maps, vehicle kinematics, clearance, slopes, and traction. Major milestones include: using explicit models of road lines, stripes, shoulders and other features to drive on highways (1989); building an integrated Local Environment Map (LEM) (1990); completion of the ACORN Core Navigation System (1990); Navigate using externally acquired maps and updating accumulated map information (1991).

VISUAL MODELING AND RECOGNITION (SRI, ADS/Stanford, GE)

This effort is concerned with developing techniques for automatically building a representation of a complex physical environment (e.g., natural terrain populated with objects such as roads, bridges, bushes and trees) based on data from imaging sensors and previously stored knowledge. The major tasks are: (1) design and implementation of a representation system that is capable of describing all the information about objects in an outdoor (natural) environment, and is capable of operating in the presence of incomplete, inaccurate, and inconsistent sensory information; (2) development of models for natural objects and terrain that are compact, permit the creation of realistic renderings, and support automatic recognition techniques; (3) development of interactive and automatic techniques for constructing models of scene objects from a priori information and sensed data; and (4) development of recognition techniques that can identify both cultural objects (e.g., buildings, roads, and fences) and natural objects (e.g., bushes, rocks, and ravines) from range and intensity data, and then update the knowledge base and the vehicle's estimate of its location.

Major milestones in visual modeling research include the development of software packages, such as the Core Knowledge System (CKS) by SRI and the system of VIEWFRAMES by ADS, and the development of testbed demonstration systems, such as the Pace system by General Electric and the TraX system by SRI. The software packages are available to other groups to facilitate their research and development efforts. For example, the CKS, which provides both spatial and semantic access to key environmental objects, is a major component of the Pace system being developed at General Electric's Corporate R&D Laboratories. The Pace system uses information from multiple sensors and uses the belief mechanisms and spatial directory of CKS to manage invalid hypotheses about physical objects. A second group at GE, Military & Data Systems Operations, has expressed interest in the CKS and is now exploring its potential for a classified project. In addition, SRI is negotiating a license agreement with Honeywell to use the CKS in target recognition, landmark recognition, and *planning applications*.

ADS has developed techniques for using rough a priori map information to direct the processing of terrestrial scenes which have been successfully applied to imagery from the ALV. ADS has also developed a theory of spatial representation, called Qualitative Navigation, which allows for robots to be simply concerned with their position and the position of landmarks. This addresses several problems in representation and navigation for outdoor, terrestrial robots, such as how to represent distant objects for which it is difficult to obtain accurate 3D information and how to navigate using limited recognition capabilities. Qualitative Navigation has been extensively tested via simulation where it has shown the ability to deal with very large positional and recognition uncertainty. Qualitative Navigation uses a viewer-centered representation called VIEWFRAMES which is based upon extracting temporally stable and visually distinctive events from a sequence of viewpoints. ADS is currently extracting this underlying representation from outdoor imagery. Another part of ADS research has been developing software environments to support research in image understanding. This includes the POWERVISION image understanding environment developed on a SYMBOLICS LISP Machine and extensions for it to run under generic CommonLisp on SUNS and MAC IIs.

The TraX system, which is being developed at SRI, builds models of objects that evolve over time as additional data are obtained from a sensor mounted on a moving vehicle. The system uses inertial navigation data to help track objects from image to image. It then builds descriptions of the objects and evaluates the stability of these descriptions over time. Since the appearance of an object can change significantly as the vehicle approaches it, a lattice of representations is provided that controls the evolution of an object's description from a crude blob to a complete semantic model, such as bush, rock, and tree. One of these descriptions is associated with an object only after the object has been detected and described multiple times and the parameters of description are stable. Stability is defined in a statistical sense augmented with a set of explanations describing reasons for missing an object or having parameters change. These explanations can invoke many types of knowledge, including the physics of the sensor, the performance of the segmentation procedure, and the reliability of the matching technique.

By the end of FY89 these representation systems will form a fully functional core for a recognition and navigation system. In FY90 and 91 plans are to extend the generality of these techniques so that they can be applied to a wide range of vision problems, including autonomous navigation for weapon systems, surveillance, and robotic manipulation and inspection.

DYNAMIC SCENE AND MOTION ANALYSIS (UMass, USC, Honeywell)

This effort is aimed at the utilization of information from a sequence of image frames obtained from a sensor undergoing arbitrary motion. Major goals are:

- recovery of the motion parameters of the sensor;
- recovery of the motion parameters of independently moving objects in the field of view;
- recovery of the depth of environmental objects and points; and
- integration of motion and depth information with environmental models in vehicular navigation systems.

While motion clearly plays an important role in vision, it remains one of the more difficult and challenging areas of experimental vision research.

The University of Southern California has developed techniques for robustly estimating 3D motion parameters and computing 3D motion trajectories of moving objects given features extracted from 2D image sequences. These techniques have been implemented in a working system for constant motion. They also allow recovery of depth information and relative structure. To support this work, USC has developed line based and region based feature extraction and matching techniques for detecting moving objects in motion sequences. In addition, USC has developed a technique for the analysis of closely-spaced image sequences that makes explicit use of occlusions and works for arbitrary observer and object motions.

The milestones for USC work in 1989 are integration of feature extraction and matching with motion estimation into a more complete analysis system that provides motion, depth and structure information for constant and simple acceleration motions. Longer term goals are inference of structure from spatio-temporal data, merging scene descriptions into global descriptions (such as those needed in a reconnaissance system), parallel implementations of motion algorithms and transfer of technology to application projects in academia, industry and Government.

Honeywell has developed qualitative reasoning and "3-1/2-D" modeling techniques for target motion detection and tracking from a moving platform in simple curved road scenes; has demonstrated that the direction of instantaneous heading of a vehicle ("Fuzzy Focus of Expansion") can be given with one degree of accuracy using image information exclusively; has shown that rotations in the horizontal and vertical directions of plus/minus five degrees or larger can be tolerated by the Fuzzy FOE algorithm, and has shown the dynamic model matching concept for landmark recognition from a moving platform. In addition, Honeywell has done initial experiments in digital map integrated target tracking

These projects are developing techniques for robust detection, tracking, and recognition of stationary and moving targets in images, as well as the depth and location of environmental objects in the scene for landmark recognition and obstacle avoidance. The images may be generated from either a stationary or moving vehicle. The motion parameters of the vehicle will be recovered passively by analyzing image motion of points and features across a sequence of images. This is necessary in order to recover three-dimensional information about environmental depth and moving objects.

Results from motion analysis will provide sensor and object motion parameters and depth data for subsequent knowledge-based processing. The dynamic event perception achieved using this knowledge will be useful for vision controlled navigation/guidance of a vehicle, as well as surveillance and other subsequent applications of military relevance. The software being developed will be kept compatible with the CMU blackboard systems so that there is a natural integration mechanism. Initially, the motion analysis has focused on images from the moving vehicle in road scenes with static obstacles and moving objects (such as another vehicle). Later, techniques will be considered for dealing with complex scenes involving a wider variety of scenarios including off-road stationary and moving objects.

Major milestones are: in 1989 detection and recovery of motion of several moving objects in more complex scenes and integrated motion cues for recognition and tracking; in 1990 digital map integrated motion detection and tracking under high clutter and maneuvering situations; and in 1991 demonstration of identifying military significant events in more complex scenes by combining perceptual cues, reasoning and expectations, and spatial and temporal analysis of dynamically changing scenes viewed from moving sensors.

Using the Martin Marietta ALV, UMass has collected several motion sequences with accurate ground truth using both stop-and-shoot and move-and-shoot scenarios for image capture. In order to support quantitative experiments in motion analysis, the ALV's land navigation system was used to obtain sensor motion parameters. In addition, the location of key environmental objects and features appearing in the scenes were obtained by means of a careful cartographic survey.

Problems with respect to the extraction of motion and depth information using traditional optical flow techniques have led us toward the exploration of methods for combining the local flow/displacement fields with larger token-like structures. To provide a more robust estimate of motion and depth parameters, methods for computing the temporal correspondence between straight line segments and the changes in the lengths between virtual intersections were developed and tested on several motion sequences. Experimental results were obtained of less than 5% error in depth for objects at a distance of 20 to 80 feet.

Goals for 1989 involve demonstration of the recovery of environmental depth by tracking straight lines; the determination of the robustness of integrating stereo and motion processing to extract occlusion boundaries and, consequently, more accurate depth results; and the demonstration of a model-directed navigation system for an outdoor mobile robot.

OBSTACLE DETECTION AND AVOIDANCE (Hughes and U.Penn)

Hughes developed an algorithm using the Markov Random Field (MRF) formalism to perform segmentation and smoothing of color imagery. The MRF approach allowed us to define a local energy function which consisted of two parts, one describing the interaction potential between neighbors, and the other associated with the difference between the predicted image and the observed data. Within the MRF framework, two processes were defined: the line process which governs the formation of discontinuities consisting of horizontal and vertical components, and the color process which performs smoothing where discontinuities do not exist. Since the goal was to compute an accurate segmentation based on local energy measures, the algorithm finds the MRF state that maximizes the probability of that segmentation which is equivalent to minimizing the energy function over the image. Hughes performed the minimization using a Hopfield network with four contributing terms: a smoothing term, a data term, a potential energy term for the line processes, and a gain term. A theory of parameter interrelationships is being developed. Experiments have included hexagonal and triangular lattices for the MRF in order to model natural terrain more accurately than the standard rectangular grid allowed.

Hughes also investigated methods to improve the computation of optical flow for estimation of general camera motion. An algorithm was implemented to obtain the optical flow at each pixel by minimizing the change in image brightness under smoothness constraints. Preliminary experiments were performed using the optical flow to recover camera rotation by minimizing the least squares error in the rotational parameters of the observed motion field. Natural terrain imagery was used for these experiments. The results appear to be accurate for cases with small distances between the images. This algorithm was incorporated into an automated topographical terrain information collector (ATTIC) system. Estimation of displacement in the latitude/longitude is provided by the local navigation system (LNS) onboard the vehicle. Previous data is used to estimate the elevation change. The ATTIC is now providing the ability to fuse sequences of sensed 3D data into a single representation for a path of over 100 scanned images and 700 meters of outdoor natural terrain travel. It is a data representation that provides an integrated memory of previously sensed data by eliminating discrepancies. It may also be useful as a mechanism to validate or update digital terrain maps.

At the University of Pennsylvania, the research emphasis has been on multisensory control and integration techniques to improve object detection and recognition. Theoretical results are beginning to provide new insights for systematic integration policies. The use of superquadrics to determine surface parameters offers natural descriptions of 3D objects useful for grasping and obstacle avoidance. Sensor models, including the physics, noise and other limitations, have been developed to predict quality and reliability of sensory data. Research at UPenn, using wavelet decomposition of a visual signal has shown that the signal can be decomposed into spatial resolutions of power 2 without loss of information.

The University of Pennsylvania will continue researching active perception, including development of control strategies with feedback for looking and feeling, the integration of perceptual information into a coherent framework, and the development of criterion for quality and quantity of information that the system must seek. Towards this goal, three tasks have been chosen. The first task will investigate scene segmentation with a feedback strategy combining multiple resolution edge detection (using wavelet decomposition) and local region growing methods. The second task will extend the complexity of 3D shape that the superquadric model can describe. The final task will investigate the spectral properties of the visual signal in order to detect color constancy.

PARALLEL COMPUTING ENVIRONMENTS FOR VISION (Maryland, MIT, CMU, Rochester, Columbia)

The University of Maryland has been studying how to effectively use the Connection Machine to solve problems in visual navigation. This research has proceeded along two directions – one concerned with efficiently multiresolution and variable resolution image processing and the second concerned with problems associated with integrating modules across all levels of visual information processing in the context of a difficult visual navigation problem.

Two alternative approaches have been developed for effectively using a massively parallel hypercube for performing multiresolution image analysis – fat pyramids and redundant pyramids. In the fat pyramid, a hypercube of processors is used to represent a pixel at any resolution level, with the size of the hypercube increasing as image resolution decreases. In the redundant pyramid, multiple copies of images are stored at each level of resolution, with more copies being stored as image resolution decreases. For both fat and redundant pyramids, algorithms have been developed for performing basic image analysis operations such as histogramming, table look-ups and convolution. Many of these algorithms have been implemented on the Connection Machine. Methods have also been developed for representing quadtrees within the Connection Machine, and a hidden surface algorithm has been designed and implemented on the Connection Machine based on this representation.

Maryland is also concerned with problems associated with integrating vision modules at all levels of processing on the Connection Machine. Project RAMBO (Robots Acting on Moving BOdies) involves navigating one robot using vision so that it can point a laser designator at a set of sensors on a target moved through space (or along a surface) by a second robot. A visual navigation system is being constructed at Maryland for solving such navigation problems. Progress to date includes the implementation of the low-level vision system, a sophisticated 3-D pose estimation program, and motion planning algorithms all running together on the Connection Machine.

MIT had concentrated in using the Connection Machine to explore parallel models of computation and for developing a vision system for unconstrained environments -- the Vision Machine. MIT demonstrated for the first time the whole Vision Machine system working from images to recognition through integration of other cues. MIT has also continued to study the performance of alternative, nonconventional architectures for navigation.

The Vision Machine is a computer system that integrates several vision cues to achieve high performance in unstructured environments for the tasks of recognition and navigation. It is also a test-bed for the theoretical progress in low- and high-level vision algorithms, their parallel implementation and their integration. The Vision Machine consists of a movable two-camera Eye-Head system -- the input device -- and a Connection Machine -- the main computational engine. Several parallel early vision algorithms which compute edge detection, stereo, motion, texture and surface color in close to real-time have been developed and implemented. They have been integrated using the technique of coupled Markov Random Field models to provide a cartoon-like map of the discontinuities in the scene. In recent months work has been done to obtain a partial labeling of the brightness edges in terms of their physical origin. As planned, the output of the integration stage has been interfaced with a model-based parallel recognition algorithm. MIT has also begun a project, together with other faculty in the EE Department, with non-DARPA funding, to develop analog and hybrid VLSI implementations of the Vision Machine main components.

Rochester has demonstrated SIMD-like programs on the BBN Butterfly Parallel Processor that show linear parallel speedup. Over the past year a hybrid architecture involving pipelined and MIMD parallelism has been developed and integrated with a high performance 9 degree of freedom robot head. Many applications for the pipeline (including tracking, color histogramming, feature detection, frame-rate depth maps, frame-rate time-to-collision maps, large-scale correlations, segmentation using motion blur, and others) have been written. The efficacy of intimate cooperation between vision computations and controlled motion has been demonstrated. Utilities to ease the programming burden for the pipeline are being designed.

Like many laboratories, Rochester plans to have a high-bandwidth interface between low-level vision processing (the frame-rate pipeline) and a powerful symbolic computing engine (a large MIMD computer like the BBN Butterfly Plus). Programming MIMD applications is difficult, and Rochester is a leader in developing operating systems (PSYCHE), performance monitoring (PPUTTS) and debugging (INSTANT REPLAY) tools to make the job easier. The Psyche operating system will support the real-time demands of dynamic computer vision and high-level planning simultaneously. The development tools provide a graphical and LISP interface to a multi-process, multi-processor application that allow repeatable single-stepping, statistics, symbolic debugging, and other "traditional" debugging techniques that have not previously been available to parallel programmers.

Rochester has implemented object recognition algorithms in neural nets, and developed hardware realizations for the resulting constraint-propagation networks. The domain includes large sets of objects, and uses Bayesian techniques to handle partial and incomplete information.

Carnegie Mellon has concentrated on developing the Apply programming language for low-level vision, and supporting applications of the Warp machine at various sites. The development of Apply has gone in two directions: mapping Apply onto new parallel architectures, and extending Apply's functionality on existing architectures, particularly Warp. As Warp is now in use at several sites, we have aided in the development of new Warp applications, including those using Apply.

Apply has been mapped onto a wide variety of computer architectures, including reconfigurable arrays of transputers and the Meiko Computing Surface; the Carnegie Mellon Scan Line Array Processor; FT Warp, a two-dimensional fault-tolerant array of Warp cells; iWarp, the VLSI implementation of Warp jointly being developed by Carnegie Mellon and Intel Corporation, and the Hughes Aircraft Corporation Hierarchical Bus Architecture. An implementation of Apply on the University of Massachusetts Image Understanding Architecture is underway. In addition, we have corresponded on possible implementation of Apply with people at the Naval Ocean Systems Command and Stanford University. In the current implementations, Apply has been used for programming, debugging of hardware and hardware simulations, and performance comparisons.

Columbia has developed a tree and pyramid machine emulator that runs on the Connection Machine; the embedding makes optimal use of the processors and communication paths. Basic parallel Lisp primitive functions have allowed rapid portability of new and existing tree and pyramid algorithms for texture, stereo, and surface interpolation systems.

Columbia has demonstrated several systems in middle-level vision in this environment or its simulations. A new statistics-based autocorrelation algorithm for surface orientation at times exceeds human performance. A second system for calculating surface properties from sparse data has demonstrated the advantages of two new provably optimal algorithms, and has experimentally documented the degrees of tradeoff between its computation and communication costs in a Connection Machine-like environment. A third system, with a coarser degree of parallelism, demonstrates the effectiveness of fusing several textural cues, and of fusing textural with stereo cues.

An important factor in this effort has been the new DARPA Integrated Image Understanding Benchmark exercise. The new benchmark is the successor to the 1986 benchmark exercise, and tests some of the same aspects of machine performance, but within the framework of an integrated vision task. The benchmark, which was developed by the University of Massachusetts and the University of Maryland involves model-based recognition of a 2-1/2 dimensional "mobile" given artificially generated depth and intensity images. The task is designed to test low- and intermediate-level processing and communication between subtasks, including top-down control of low-level processing. The University of Massachusetts developed a sequential solution to the problem, and a sample parallel solution, which were distributed along with test data sets to over 25 vendors and research groups. In October of 1988, a workshop was held where complete results were presented for the Sequent Symmetry-81, Alliant FX-80, Carnegie-Mellon WARP, and Sun-3 and Sun-4 workstations. Partial results were presented for the Thinking Machines Connection Machine, and the Intel IPSC-2. Complete simulation results were also presented for the ASPEX ASP, and the University of Massachusetts/Hughes Image Understanding Architecture (IUA). Other vendors and research groups are continuing to develop their own implementations of the benchmark. Several benchmark extensions and improvements were suggested by workshop participants, many of which will be incorporated in a future version of the specification.

TECHNOLOGY TRANSFER (Maryland et al.)

Martin Marietta Corporation (MMC) at Denver, and the Computer Vision Laboratory at the University of Maryland (UM) collaborated on a study comparing methods for reconstructing the 3D shape of roads. One method uses video images from a single camera, while the other combines video data with images obtained by a range scanner. These methods can be used for the autonomous steering of a vehicle. The data used for the comparisons were actual images collected by the Autonomous Land Vehicle (ALV). The methods are robust and gave consistent interpretations for the 50 road configurations tested. The MMC method is potentially more powerful, and would make other methods useless if the range scanner was able to cover the same field as a video camera. With a limited range scanner, however, the methods appear complementary if they are combined so that the MMC method provides the ground truth close to the vehicle, and the UM method uses this ground truth to extend the 3D reconstruction to most of the visible road.

Several sites are using Warp for applications, particularly in the Strategic Computing, ADRIES, and SCORPIUS programs. Warp is being used regularly in the Strategic Computing program at Carnegie Mellon for control of NAVLAB, including implementation of the SCARF color road-following algorithm. Apply and WEB routines have been used in the ADRIES program at Science Applications International Corporation, and in the SCORPIUS program at Hughes Aircraft Corporation, for tasks involving interpretation of synthetic aperture radar images. Researchers at Martin Marietta used Apply and WEB in the Autonomous Land Vehicles program, and Apply and WEB have been used at ESL Corporation in various programs. Much of this was reported at the Warp User's Group held in Cherry Hill, New Jersey, 20-21 June 1988, sponsored by GE Aerospace Labs, Moorestown, New Jersey.

Honeywell is applying SCVision technology developed under the DARPA Scene Dynamics program for:

- Vision-based obstacle detection during rotorcraft low-altitude flight program from NASA.
- Model acquisition and refinement program from a classified agency
- Glide Bomb Unit, GBU-15 trainer program from Air Force

Hughes autonomous navigation simulation package is a potential development and evaluation tool being considered for the JPL/NASA Mars rover program.

Rochester's Butterfly software is disseminated through BBN, and is freely available. There is evidence that scientific papers have transferred some of the technology successfully. Through an international computer newsgroup the expertise on the DataCube pipelined processor is both shared and acquired.

ECRC (European Computer Industry Research Center) has implemented Instant Replay on an [illegible] processor Siemens MX500 (their copy of the Sequent). The idea has also been incorporated in the Amoeba debugger (Amoeba is a distributed system used widely throughout Europe).

In addition to the usual transfer of technology via publications (papers published in Science, Proc IEEE, etc.), MIT has provided software to other CM users, including ADS, Maryland, Xerox PARC, Hughes Aircraft, Perkins-Elmer, and TMC. MIT has also collaborated more directly with Hughes, with an exchange of people in both directions. MIT edge detector and stereo programs have been distributed widely to universities, a Navy laboratory, and defense contractors. One of MIT's (serial) recognition algorithms is used by Technical Arts in Redmond, Washington for applications for Boeing.

CMU, which is building the NGS vision system, has sent to Martin Marietta the NGS Blackboard, the Generalized Image Library image data package, programs for analyzing range data from the ERIM using the WARP computer, and the WEB and APPLY image processing tools for the WARP. The CMU program for building 3-D maps and following the road using reflectance data drove the ALV in August 1988. Hughes has performed a successful demonstration of cross-country navigation and obstacle avoidance on the ALV. In addition, the CMU NGS Blackboard has been sent to other ALV contractors including ADS and FMC.

The NGS Blackboard has been exported to several non-DARPA sites, including NASA-Goddard, DEC, and Florida Atlantic University (for use in underwater robot design). The 3-D mapping developed at CMU has been used extensively in designing and building the AMBLER, a prototype Mars Rover walking robot sponsored by NASA. Other NAVLAB technology has been used by a number of other projects, including autonomous navigation of heavy construction equipment.

PROGRAM EVOLUTION

The NGS will continue to evolve as one of the products of SCVision research. Technology transfer to the NGS from other contractors has been, and will continue to be, mostly through papers and discussions of results. Similarly, the impact of the NGS on applications systems will be in terms of ideas demonstrated and technology developed, rather than in terms of actual code delivered. The perceptual algorithms and architecture of the NGS have been developed for a land vehicle, and should be directly applicable to the Army's RCC program. The technology is to a large extent also applicable to guiding unmanned air vehicles (PIONEER, AMBER); assisting a human pilot (Pilot's Associate); and unmanned underwater vehicles (MAUVE). Non-DOD applications include a strong influence on NASA's Mars Rover program. A particularly important future application of autonomous land vehicles will be characterization and cleanup of hazardous waste sites. The amount of work to be done is enormous, and performing the tasks by humans is very dangerous. The cross-country mapping and traversal capabilities being developed and demonstrated as part of the NGS are ideal building blocks for future cleanup vehicles.

Building capable mobile robots requires all of the technologies being developed as part of SCVision: modelling and recognition, dynamic scene and motion analysis, obstacle detection and avoidance, and parallel computing environments. Combining the individual pieces into working systems requires the systems software of the New Generation System, and the real-world testing only available with outdoor mobile robots. The work to date has solved some of the problems, has developed approaches to other problems that work in limited cases or at slow speeds, and has uncovered further research topics. Continued research, focused on outdoor autonomous navigation, will put us at the threshold of being able to build viable real-world intelligent systems.

The techniques being developed in SCVision for representing, recognizing, and reasoning about natural and man-made objects have applications in many other areas, including cartography, photointerpretation, and tactical target recognition. A natural evolutionary step for this program would be to explore these applications. In cartography, for example, current stereo mapping techniques work well on downward-looking images from significant elevations. However, they have trouble with high obliques of scenes containing buildings, tree cover, and high relief features, because the matching algorithms do not "understand" the three-dimensional nature of these things. In target recognition the pattern recognition techniques explored to date have had very limited success for similar reasons.

Acknowledgements

This update was compiled from the contributions of all the participants of the SCVision program. I especially want to recognize Chuck Thorpe and Takeo Kanade for their work in helping to synthesize the various contributions from the SCVision community into the initial draft. I am solely responsible for any oversights and subsequent garbling of the original content.

SPATIAL UNDERSTANDING: THE SUCCESSOR SYSTEM

Thomas O. Binford

Robotics Laboratory
Department of Computer Science
Stanford University, CA 94305

ABSTRACT

Progress in general, model-based vision is presented. This reports includes a summary of the SUCCESSOR system with updates. The objectives are multi-sensor interpretation and stereo mapping. A zeroth release of SUCCESSOR has been made. SUCCESSOR is implemented in Commonlisp with device independent graphics and user interface to enhance portability. Extensions to the modeling system were completed for the release. Extensive developments have been made in interpretation using Bayesian networks which exploit networks built up from object and scene models. Recent contributions have been made in perceptual strategies for efficient computation in interpretation, including distributing computation among parallel processors, knowledge acquisition and resource allocation. An experiment is described in recognizing a class of objects from their image contours. Segmentation into color regions is described. Progress has been made on curve linking to make extended edges.

1. INTRODUCTION

Model-based vision means different things to different people. We mean general, model-based vision which includes two challenges. One is to provide general methods to use simplifications for special cases, e.g. few objects, simple environment, or limited range of viewpoints. These methods are valuable for applications in limited environments. The second challenge is to use models in extremely varied situations and objects , e.g. out-of-doors with trees.

Some objectives of SUCCESSOR are: multi-sensor integration; fundamental vs phenomenological modeling; comprehensive probability models in interpretation.

A theme in SUCCESSOR is fundamental modeling of the total system. Geometric modeling is one aspect of system modeling. Modeling of the perceptual system includes modeling of sensed images and modeling perceived structures from images, i.e. modeling of sensors, modeling of illumination or sources, modeling the interaction of surfaces with light and other signals, and modeling perceptual operators.

We consider vision in a complex, three-dimensional world, e.g. the outdoors. The outdoors has great variability. Trees, bushes, rocks and terrain have great variation in structure and in appearance. There are large numbers of different objects with complex surface markings in complex environments. In industrial applications, problems can often be engineered to be simple. We have a commitment, that general systems can be built up from general purpose components and that many applications will be made possible and effective using specializations of these components together with some general mechanisms for building and using application-specific operations and knowledge.

The zeroth level release of SUCCESSOR was made in July 1988. The release is about 50,000 lines of Commonlisp code. Relatively extensive documentation was completed by our standards, although informal by higher software standards. An automated documentation facility was implemented. A port of part of SUCCESSOR has been made to the Mac II by Chelberg. We are about to port parts of SUCCESSOR to the Silicon Graphics Personal Iris. A port to a SUN is underway by Michael Black of ADS. We are now evaluating portable "make" facilities provided by ADS to simplify porting. Probably the most difficult parts to make machine-independent are graphics and user interface modules. These and development environments are not specified in the Commonlisp standard. To maintain portability, we have defined a device independent graphics module, DVI, which includes an elementary user interface module with menus. We are looking now at X-11 as a standard to integrate with DVI.

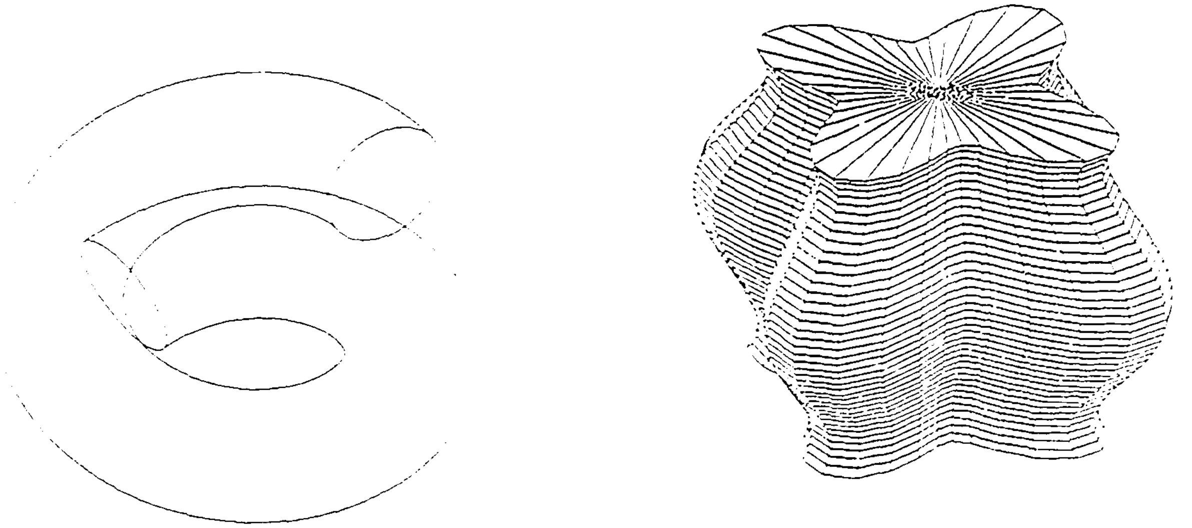


Figure 1. CSR and SHGC forms
Curved Solids of Revolution and Straight Homogeneous Generalized Cylinders are the primitives volumes in the CSG modeling system for SUCCESSOR.

2. MODELING SYSTEM

The modeling system was extensively rewritten for the release. Curved Surfaces of Revolution (CSR) were integrated into the system along with Straight Homogeneous Generalized Cylinders. See figure 1. Curved Surfaces of Revolution are generalized cylinders with constant, circular cross section, i.e. constant sweeping rule along a curve. The representation and intersection code were extended greatly to include CSRs and cleaned up. The 2D modeling system for modeling cross sections was improved and generalized to allow arbitrary spline bases for each piece and to allow arbitrary specification of continuity or discontinuity conditions. The implementation remains tied to (r, θ) single-valued representation, i.e. star-shaped. Preliminary work has been done to generalize the 2D modeling system to use ribbons, i.e. to generalize descriptions of cross sections to include part/whole graphs with ribbon primitives (generalized cylinders in 2D). Some work is underway to implement non-uniform rational B-splines (NURBs). Work on surface representation continues.

Primitives are constructed with a menu-driven geometric editor similar to MacDraw which aids in constructing generalized cylinders by defining their cross sections and sweeping rules as piecewise smooth curves, i.e. splines. The editor has three windows, cross section spline, sweeping rule, and primitive solid, which are shown in figure 2.

CSG parts models are built with set operations of CSR and SHGC primitives [Ponce and Healey 88]. Figure 3 shows an example of the intersection of generalized cylinder primitives. Surfaces of generalized cylinders are defined by two parameters, (θ, z). The parameter space is searched by a quadtree; a box tree in 3-space provides a nested set of enclosing volumes corresponding to the quadtree. The hierarchical search is guaranteed to find intersections, it is relatively efficient, and is carried out to relatively high accuracy. Times required are of order 10 minutes on a Symbolics 3600 with Commonlisp code that is not heavily optimized. The resulting intersections are made consistent by a method which exhausts degenerate cases of polyhedral intersections. Intersections are computed on the parameter space to form trimmed surface patches.

Compound parts and assemblies are defined by affixment operations which define transformations among coordinate frames tied to parts. Affixments are parameterized symbolic expressions which may involve time, e.g. in animmtion. Assemblies are defined by a small language. They are most conveniently built with a geometric editor.

There is a back to front painting algorithm for polygons which does not require a z-buffer. It can be very efficient on computers that have fast polygon painters. The method does not apply to composite objects.

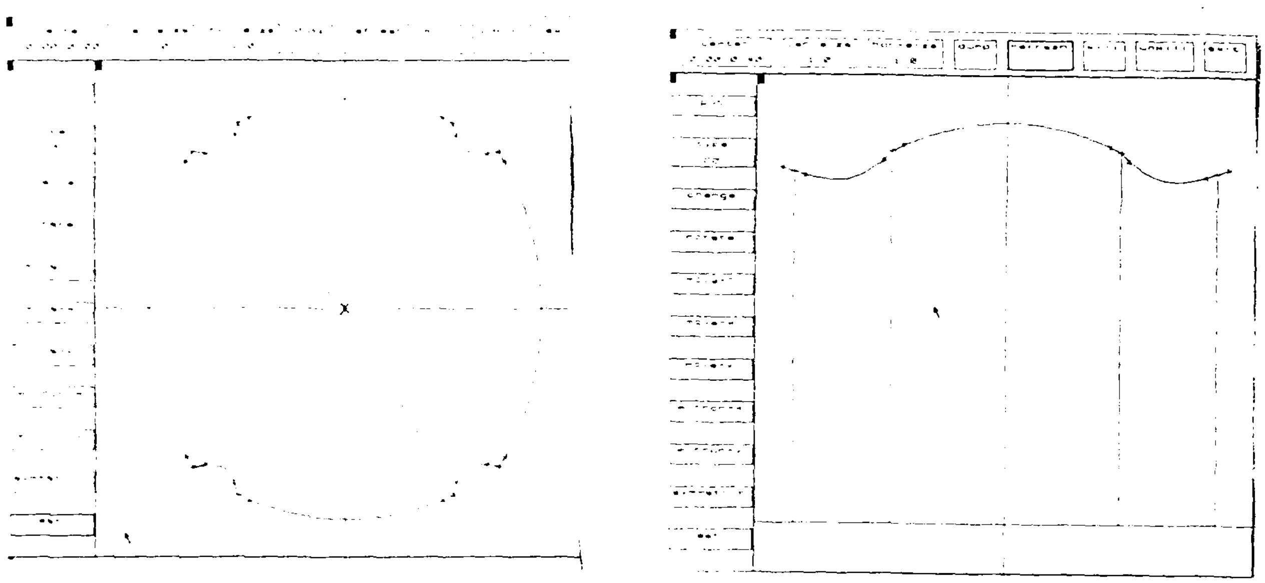


Figure 2. Geometric Editor
Generalized Cylinder primitives are constructed in a menu-driven geometric editor.

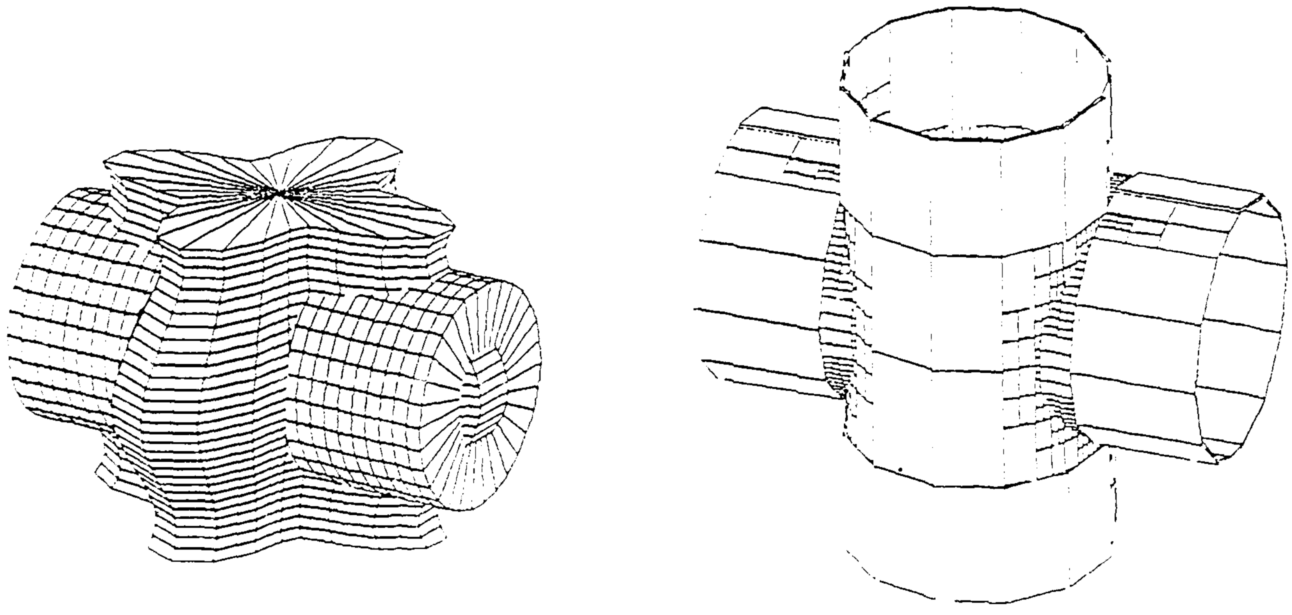


Figure 3. Set Operations
Line drawing

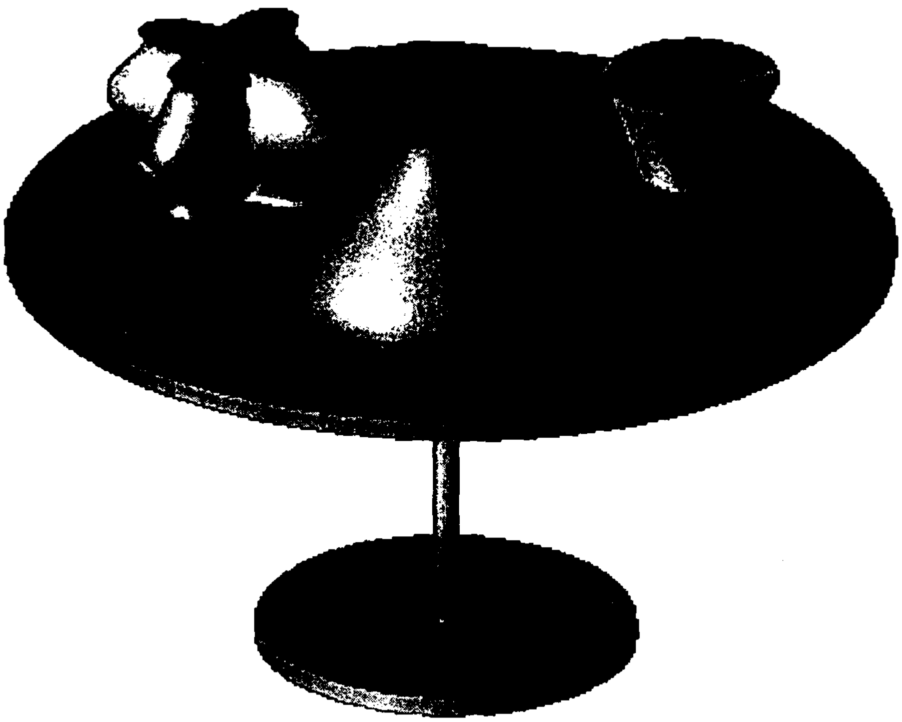


Figure 4. Shaded rendering.

but is useful in the interactive primitive editor.

Z-buffer rendering of polygons is slower than the painting algorithm, but faster than shaded ray tracing, and is used to generate most of our shaded pictures. Computing time is about 20 minutes for complex models. Shaded ray tracing remains the most flexible and realistic rendering technique. Here, we again use the box tree representation of the surface patches to get acceptable computing times. Individual rays are intersected with the tree associated with a given surface patch. If the ray intersects the corresponding box, the box is subdivided, and the recursion proceeds. Otherwise the subdivision stops, there is no intersection. Shadows and texture mapping have been implemented. The complexity is $O(N.q)$ where N is the number of pixels, and q is the depth of the box trees considered. It is much better than the $O(N.4^q)$ of classical algorithms, but computing time for complex pictures remains long: several hours.

A variant of ray tracing is done for line drawing display. We compute limbs and edges as for wireframe display, and do ray tracing only at contour pixels. This method reduces the complexity of ray tracing by a factor of S/P where S is the total projected area of the objects drawn, and P is the total projected perimeter of their contours. Complexity is roughly $O(\sqrt{N}.q)$ (area grows as the square of perimeter when resolution increases). This method is relatively fast (about 5 minutes for the elbow).

Graphics is primarily of use for researchers to understand the system and to demonstrate results of the system. Graphics is one method for modeling images, i.e. modeling appearances of objects. Graphics methods are useful for individual objects from individual points of view. A challenge in model-based vision is modeling images in a much more general way, for example for classes of objects like trees for unknown viewpoints in complex environments, like the outdoors.

A theme in SUCCESSOR is fundamental modeling of the total system. Geometric modeling is one aspect of system modeling. Modeling of the perceptual system includes modeling of sensed images and modeling perceived structures from images, i.e. modeling of sensors, modeling of illumination or sources, modeling the interaction of surfaces with light and other signals, and modeling perceptual operators.

Material modeling has been included in the modeling system [Healey 87a]. Detailed and generic models of surface reflectance have been included especially in the visible spectrum. Phenomenological models from physics have been incorporate for both metals and non-metals for specular and diffuse reflectivity. [Healey 89b] quantifies and supports the usual assumption made in computer vision and graphics that spectral properties are independent of geometry. This argument supports the use of normalized color under controlled circumstances in image analysis. Briefly, the Torrance and Sparrow phenomenological model quantifies specular reflection, surface reflection. Metals have only surface reflection. The Reichman-Kubelka-Munk model is familiar in psychology, but had not been used in computer vision. It is a phenomenological model which quantifies body reflection, diffuse reflectivity from the interior of dielectrics. Non-metals have a mixture

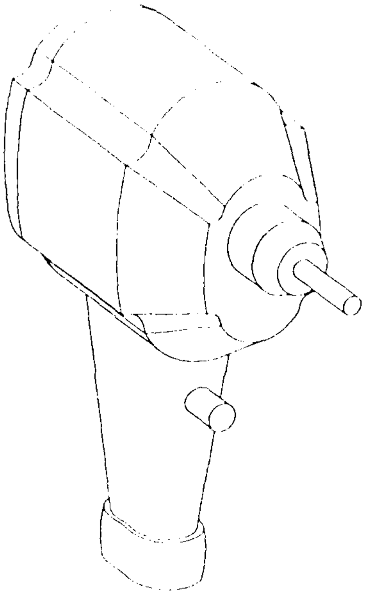


Figure 5. The ray traced drill.

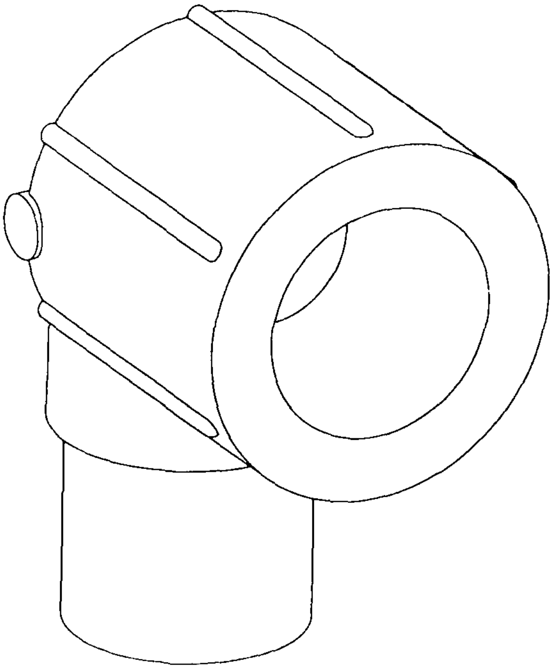


Figure 6. The ray traced elbow.

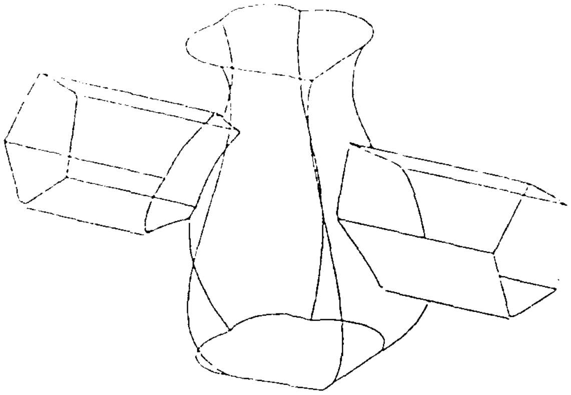


Figure 7. VSCP graph
An exploded drawing of surfaces of an object shows components of the Volume, Surface, Curve and Point graph of object topology.

of surface and body reflection, diffuse and specular reflection. In a later section we discuss segmentation using color.

Preliminary experiments with new generic modeling capabilities were described at the last IU Workshop [Kriegman, Binford, Sumanaweera 88]. *That system has been developed further. Although we plan to* extend generic modeling greatly, generic models will be used for several new efforts, namely for generic prediction, for functional specification, and for a new constraint system which like ACRONYM will include symbolic expressions with variables which may be partially speciafied. However, the new system incorporates probabilistic variables and constraints.

The topological structure of object models is being incorporated in a VSCP graph of volumes, surfaces, curves, and points, the topological types of dimension 3, 2, 1, and 0 in 3-space. It is also called the BFEV graph, terminology from the blocks world which was used in the Stanford Hand-Eye system in 1970. The graph describes fundamental spatial relations in 3-space. It is not an aspect graph which describes phenomenological relations (appearance in 2-D projections). Algorithms determine the appropriate connectivity and adjacency structure of the VSCP graph automatically from object models, and allow cutting and pasting. Figure 7 displays an example. The analysis determines surface continuity automatically, e.g. boundaries of C^1 and C^2 patches with tangent and curvature continuity and maintains continuity in the structure. The VSCP has natural bounding relationships: surfaces bound volumes; curves bound surfaces. Each body, surface, and edge has a local parameter space domain, mapping function (to map the 2D domain onto the 3D range) and reference coordinate frame. Curves correspond to geometric discontinuities observable generically in the image (i.e. except on a compact set of measure zero) [Binford 87]. The surface discontinuities arise from SHGC termination conditions, discontinuities in the cross section and sweeping rule functions, and the intersection of SHGC's in composite models.

VSCP graphs are hierarchical; they can be used flat if needed. A single C^0 surface includes the surface of a body (possibly several). C^1 and C^2 patches correspond to different observables in an intensity image or range imagee. Physical properties of the surface such as reflectance or texture can be examined on this surface.

3. INTERPRETATION

We address issues of *effective recognition, multi-sensor interpretation, and low complexity computation.* SUCCESSOR has fundamental modeling of objects, their geometry and physics, rather than phenomenological modeling of appearances. These physical models are directed acyclic graphs of part/whole relations. As described above, the graph of volumes linked by CSG set operations and assembly operations is included in the VSCP graph which also includes surfaces, curves, and points in space. The VSCP model graph has generic (class) and individual models.

Segmentation of direct depth data gives surfaces, edges, vertices, and volumes which correspond directly to topological entities in space in the VSCP graph of models. For image intensity data, SUCCESSOR integrates and interprets connected image components as connected components in space, linking parts into objects [Binford 81]. The primary interpretation in SUCCESSOR is 3-space, although image-based matching is included. There are varied images available, intensity boundaries, color, shading, texture, stereo, motion, direct depth measurement, infrared images, radar, and others. There is also non-image information from knowledge of objects, behavior, and context.

The VSCP model graph has typically a half dozen levels or so. The graph of observables has a similar number. Data, information, and knowledge occur at a dozen or more levels. In combining multi-sensor data, each type of measurement or information is represented naturally and accurately at a corresponding level in the network. We maintain natural problem relations in a network containing geometrical and physical constraints with models of uncertainty in measurement and information [Binford, Levitt, Mann 87]. In ACRONYM, inequality constraints were the basis for interpretation [Brooks 81]. SUCCESSOR extends this greatly to include a hierarchical network of relations among constraints and to introduce evidential reasoning under uncertainty.

In [Nevatia 74] we demonstrated a low complexity method for effective recognition by building coarse, stick figure descriptions to use for indexing, i.e. generating hypotheses for subclasses of objects with similar topology. This was estimated to be effective for abc 1000 objects. We have used quasi-invariants for indexing, for hypothesis generation of a small number of candidate hypotheses [Binford, Levitt, Mann 87, Arnold and Binford 80].

We emphasize low complexity methods in which the Bayesian network is partially instantiated dynamically, from the VSCP graph, instead of generating a complete static interpretation network in advance.

In new work, [Levitt, Binford, and Ettinger 88] developed a utility-based analysis to generate efficient control of interpretation. An essential problem here is the distribution of high level perceptual operations over parallel processors. Work is underway to formulate this in an influence diagram framework. We have begun work to use utility-based control in generating perceptual strategies for visual navigation by our mobile robot.

Objectives for an interpretation system can be classified along multiple axes: many or few objects in an image; many or few objects or classes possible; generic object class or individual, identical objects; generic observation or not, i.e. with viewpoint-insensitive or viewpoint-sensitive methods. A majority of the community choose few, individual objects, with image-based viewpoint-sensitive, multi-view methods. In the multi-view approach, each view is a separate object. Our aim for SUCCESSOR is many objects in an image, many object classes in the world, generic object classes, and generic observations. We take a space-based, object-centered approach. Structured descriptions which can be achieved are expected to be incomplete and contain errors. The basis of the structured approach is that variation is much less in 3-d body-centered relationships than in images.

[Ponce and Kriegman 89] address the recognition and positioning of curved three-dimensional objects from their monocular image contours under the following assumptions: Precise geometric models of the observed objects (and/or object classes) are available. The data consists of imperfect edge-maps; in particular, knowledge of high level features such as junctions or corners is not required. No additional information such as shading, surface normals, or range is available.

This problem is difficult because it involves comparing the shape of the two-dimensional surfaces that bound the observed objects to the one-dimensional curves that form their contours in the image. For polyhedra, this is relatively easy, since the contour generators (edges) of these objects are view-independent. The situation is quite different fo. curved objects, whose contour generators move and deform over the surface according to the observer's position.

To make real progress, it is necessary to understand the geometry of image contours and to explicitly

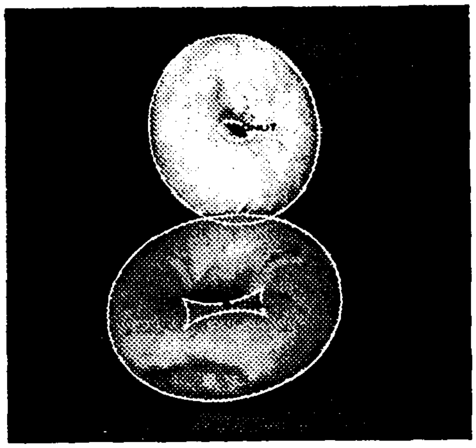


Figure 8. Recognition and positioning: a bagel vs. a doughnut.

relate their shape to the shape of the observed objects and to the viewing parameters. A new approach to that problem is proposed for object models consisting of collections of algebraic surface patches and their intersection curves. This includes nearly all representations used in computer aided design and computer vision, such as CSG models, generalized cylinders, and superquadrics. The image contours considered are the projections of surface discontinuities and occluding contours.

Elimination theory provides a method for constructing the implicit equation of the image contours of an object observed under orthographic, weak perspective, or perspective projection. This equation is parameterized by the position and orientation of the object with respect to the observer. Determining these parameters is reduced to a fitting problem between the theoretical contour and the observed data points.

Two measures of fit are proposed: The implicit equation can be directly fitted to the data points. Alternatively, elimination theory can be used to construct a closed-form expression for the distance between an image point and the theoretical contour. Position and orientation are then determined by minimizing the average of this distance over the data points. The proposed approach readily extends to parameterized models, whose contour equation simply includes additional shape parameters.

A simple recognition and positioning system has been implemented for a world composed of tori of different sizes and flavors. It has been successfully tested on several real images of objects such as plastic rings, doughnuts, and bagels (see figure 8), and has proved to be both reliable and computationally efficient.

4. SEGMENTATION

[Healey 89] presents a parallel color algorithm for image segmentation. From an input color image, the algorithm labels each pixel in the image according to one of several regions with uniform normalized color. An edge operator is first applied to the image. Rectangular regions are successively split until they contain no edge elements, i.e. they are uniform. A mean normalized color is computed. Uniform regions are assigned to matching color regions or used to create new regions. The algorithm has been demonstrated on metal objects, plastic objects with specularity, and matte surfaces. It has moderate performance and is relatively fast.

Healey has examined the phenomenological models for body reflectance and specular reflectance. He has also examined experimental data for a moderate number of surfaces in order to quantify the range of validity of the approximation that reflectance is independent of geometry. This is the usual implicit assumption in computer vision. He shows that the approximation is quite valuable, i.e. that it holds over most viewing angles.

ACKNOWLEDGEMENTS

Support for this work was provided in part by the Air Force Office of Scientific Research under contract

F33615-85-C-5106 and by the Advanced Research Projects Agency of the Department of Defense under Knowledge Based Vision ADS subcontract S1093 S-1.

REFERENCES

[Arnold and Binford 80] Arnold, R.D., Binford, T.O.; "*Geometric Constraints in Stereo Vision*"; Proc SPIE Meeting, San Diego, Cal, July 1980.

[Binford 81] T.O.Binford; "Inferring Surfaces from Images"; Artificial Intelligence Journal August, 1981.

[Binford 87] T.O.Binford; "Generic Surface Interpretation: Observability Model"; Proc Int Symp on Robotics Resear.:h, 1987.

[Brooks 81] R.A.Brooks; "Symbolic Reasoning among 3-dimensional models and 2-dimensional images"; Artificial Intelligence 17,285, 1981.

[Healey 87b] G. Healey, T.O.Binford; "The Role and Use of Color in a General Vision System"; Proc DARPA Image Understanding Workshop, 1987.

[Healey 89] G. Healey; "A Parallel Color Algorithm for Segmenting IMages of 3-D Scenes"; Proc DARPA Image Understanding Workshop, 1989.

[Healey 89b] G. Healey; "Using Color for Geometry-Insensitive Segmentation"; J. Optical Society 1989.

[Kriegman, Binford, Sumaneweera 88] D.J.Kriegman, T.O.Binford, T.Sumaneweera; "*Generic Models for Robot Navigation*"; Proc DARPA Image Understanding Workshop, 1988.

[Nevatia 74] Nevatia, R.; "Structured Descriptions of Complex Curved Objects for Recognition and Visual *Memory*"; *Artificial Intelligence* Laboratory, Stanford University, Memo AIM-250, 1974.

[Ponce and Healey 88] J. Ponce and G.Healey; "Using Generic Geometric and Physical Models for Rep*resenting Solids*"; *Proc DARPA Image Understanding* Workshop, 1988.

[Ponce 89] J.Ponce, D.Kriegman; "On Recognizing and Positioning Curved 3D Objects from Image *Contours*"; Proc DARPA Image Understanding Workshop, 1989.

IMAGE UNDERSTANDING RESEARCH AT SRI INTERNATIONAL

Martin A. Fischler and Robert C. Bolles
Artificial Intelligence Center
SRI International
333 Ravenswood Avenue
Menlo Park, California 94025

Abstract

Image understanding research at SRI International is a broad effort spanning the entire range of machine vision research. In this report we describe our progress in two programs: the first is concerned with modeling the earth's surface from aerial photographs; the second is concerned with visual interpretation for land navigation. In particular, we describe progress in the design of a core knowledge structure; in representing, recognizing, and rendering complex natural and man-made objects; in recognizing and modeling terrain features and man-made objects in image sequences; in interactive techniques for scene modeling and scene generation; in automated detection and delineation of cultural objects in aerial imagery; and in automated terrain modeling from aerial imagery.

Introduction

The overall goal of Image Understanding research at SRI International is to obtain solutions to fundamental problems in computer vision that are necessary to allow machines to model, manipulate, and understand their environment from sensor-acquired data and stored knowledge.

In this report we describe progress in two programs.[1] The first is concerned with modeling the earth's surface from aerial photographs; the second is concerned with allowing a robotic device to successfully navigate through, and interact with, a natural 3-D environment based on real-time interpretation of sensory data.

In the discussion of the first program we describe our progress in developing techniques for automated terrain modeling from aerial imagery; automated detection and delineation of cultural objects in aerial imagery; and interactive techniques for scene modeling and scene generation.

In the discussion of the second program we describe progress in developing techniques for automated real-time recognition of terrain features and man-made objects from image sequences acquired by a combination of ranging and photographic sensors.

Common to both programs, we describe progress in developing new techniques for representing, recognizing, and rendering complex natural and man-made objects; and the construction of a core knowledge structure (CKS), which can serve as the integrating mechanism for a new generation of generic vision systems. These systems will be knowledge-base driven, rather than task-specific (using techniques in which domain knowledge is compiled into the interpretative algorithms).

An important theme in much of our current work is an emphisis on computational performance — especially through the development of algorithms capable of exploiting the new parallel machine architectures now available (e.g., the Connection Machine™).

Design of a Core Knowledge Structure

The natural outdoor environment poses significant obstacles to the design and successful integration of the interpretation, planning, navigational, and control functions of a general-purpose vision system. Many of these functions cannot yet be performed at a level of competence and reliability necessary to satisfy the needs of an autonomous robotic device. Part of the problem lies in the inability of available techniques, especially those involved in

[1] Supported by the Defense Mapping Agency and Defense Advanced Research Projects Agency under contracts MDA903-86-C-0084 and DACA76-85-C-0004.

sensory interpretation, to use contextual information and stored knowledge in recognizing objects and environmental features. Our goal in this effort, described in a previous paper [Smith&Strat87], is to design a core knowledge structure that can support a new generation of knowledge-based generic vision systems.

A key scientific problem we address in this task is how to devise a way of describing the appearance and characteristics of any given physical environment so thoroughly that we are assured that deficiencies in available vision techniques can be overcome by access to such prior knowledge. We cannot resort to the equivalent of using a pixel-level description as the only or ultimate solution because such detailed data would be impractical to obtain, store, retrieve, or use in the interpretive process; such low-level data would almost certainly be inaccurate when obtained, and would quickly degrade as physical changes occur. (Even illumination changes would cause a pixel-level description of appearance to become useless.) Finally, accurate interpretation must be based on more than just image appearance, and there is no immediately obvious way of describing and storing such semantic (nonpictorial) information at arbitrary levels of detail.

The CKS is designed as a community of independent interacting processes that cooperate in achieving the goals of the scene modeling system. These processes may represent sensors, interpreters, controllers, user interface drivers, or any other information processor. Each process can be both a producer and a consumer of information. Each has access to, and control over, a certain limited portion of the knowledge/database resources. The CKS architecture permits access to stored knowledge by both geographic location and by semantic content.

An initial implementation of the CKS was completed early in 1987, and subsequent activity has had two objectives: applying the CKS to a variety of distinct perception problems, and improving and extending its capabilities. The most significant recent enhancements to the CKS have been the design and implementation of a temporal directory as an index to dynamic data, and the coupling of the CKS to the Cartographic Modeling Environment (discussed later, also see [Hanson&Quam88].

There are currently several efforts underway that make use of CKS as part of a perceptual system. Corporate Research and Development at General Electric Company is developing an intelligence analysis system called Pace, which includes CKS as a major component [Corby88]. It fuses information from multiple sensors and uses the belief mechanisms and spatial directory of CKS to manage hypotheses about physical objects. CKS is also used routinely in our own research at SRI. One of these efforts, which uses the context provided by the CKS database to constrain visual processing, is briefly described below.

A Machine Vision System Built on the CKS

The Core Knowledge System was designed to serve as the central information manager for a sensor-based autonomous system. In one of the robotic vision research efforts we describe in these proceedings [Fischler&Strat89], we address the problem of designing a vision system for a vehicle that can recognize objects and create a map of a piece of terrain from the imagery it collects while traversing the area. The Core Knowledge System serves two important functions in support of such a vision system. First, it encodes a world model that exists beyond the interpretation of a single image or sequence of images. Such a persistent description of the terrain and the objects that occur there is a necessary component of a vision system that is to benefit from its accumulated experience and which needs such a model as a basis for planning future activity. Ideally, the contents of the CKS should never be emptied, rather an increasingly detailed model should be developed from the partial interpretations made by the vision system as the robot repeatedly traverses the areas of interest.

Second, the existence of a world model provides the context for image interpretation. Prior expectations of what is to be found in an image makes it easier to recognize scene features. The Contextual Vision System (CVS) we are constructing makes widespread use of the CKS for representing the world, for controlling its analysis decisions, and for checking the global consistency of its results. The CKS makes it possible for CVS to reason about the full three-dimensional nature of the environment. Its context-driven control structure is a means for avoiding the exponential complexity inherent in visual recognition of a complex world. The conclusions drawn by the vision system are stored in the CKS, and are thus immediately available to a robot's route-planning, navigation, and task-specific subsystems.

Representing the Dynamics of the Environment

During this past year, we have extended our database design and developed a methodology for representing and organizing the temporal aspects of the data. This design allows data access not only through three-dimensional spatial indices and semantic categories, but through the temporal behavior of the data as well. With this design, queries about an objects identity and location in the world can be resolved, as well as queries about the future or past locations of the object. The ability to predict an object's future location and behavior is the basis for the effective management of sensory information by preventing what would otherwise be an inundation of redundant information.

The design is based on the construction of dual space representations of the motion parameters of an object. By storing and indexing within the dual space, it can be very efficient to predict the future motion of an object, or to find all objects that could move to a given location at some point in the future. It is important to choose an appropriate parameterization if a storage efficiency or retrieval advantage is to be obtained. We have already discovered several such parameterizations for some ordinary classes of motion and may include others if the need arises. In addition to representing physical motion, the dual space approach is useful for modeling the change in attributes other than location. For example, "motion" through color space, growth of a plant, and change in air temperature are potential applications. (A report on this work is now in preparation.)

Direct Manipulation of CKS Database

Because the intended domain of the CKS is the physical, outdoor environment, it is often useful to view the contents of the database through synthesized imagery. Several interfaces have been constructed to allow for interactive retrieval and manipulation of the contents of the database.

One interface is a sophisticated interaction and display facility that has been constructed from several tools developed at SRI. This interface allows full three-dimensional rendering and manipulation of any subset of the database. The primary rendering of a landform is provided by the Cartographic Modeling Environment [Hanson&Quam88a,b] to allow viewing from any perspective; icons corresponding to each data token in the subset of interest are superimposed at their appropriate physical location. Some objects are displayed using superquadric primitives to produce images more realistic than those with stylized icons. Data tokens that have specified geometric modeling information are displayed using that information; the remainder are displayed using "prototype" models stored in the semantic network. The final result is a powerful modeling and display system, coupled to the knowledge/database, that is able to provide convincing synthetic imagery from symbolic information stored in the CKS.

Representing, Recognizing, and Rendering Complex Natural and Man-Made Objects

The main theoretical issue we address in this effort is how to model a large class of natural and man-made objects in a functionally useful way. The domain for our research is outdoor navigation, in which a robotic device starts with an initial model of its environment and incrementally updates this model (and its relative position) as it moves to gather data or perform a task. We require the device to improve its performance by increasing its ability to recognize objects and/or decreasing its processing time as it sees things over and over again.

We have partitioned this type of perception task into three stages: *model instantiation, mission planning*, and *execution*. In the instantiation stage, a user gathers as much a priori information as possible about the area of interest. This may include selections from a standard set of cartographic items, such as terrain maps, soil classification maps, and road networks. In addition, given a specific mission, the user may interactively augment this database with higher-resolution descriptions of a few key features. In the mission-planning stage, the user explores possible vehicle paths and evaluates their viability in terms of several factors, one of which is the interaction of the control system with the perception system. The planning stage provides such things as a list of expected landmarks and descriptions of their visibility and shape. In the third stage, the execution stage, the vehicle performs its mission, navigating around obstacles and updating its position estimates as it heads toward its objective.

A key to successful performance in all three stages is the set of representations used to describe the environment. An object may be sketched in the model instantiation stage, projected into synthetic images during the planning stage, and matched in the execution stage. Therefore, the representations, in addition to covering a wide variety of man-made and natural objects, must be able to express a range of abstraction and precision. Our strategy for exploring these representation issues is three-fold. First, we are developing a set of representations for classes of features, such as terrain patches, rocks, and trees. Second, we are developing a Core Knowledge System (see a previous section) to serve as an integrating mechanism for all the information about an environment. And third, we are evaluating our progress by performing experiments using real data obtained from the "red-rock" area at the Martin Marietta site outside Denver.

In September 1987, three SRI researchers spent a few days at Martin Marietta surveying prominent features in the selected area and gathering an initial set of range and intensity sequences consistent with our navigation scenario. Martin Marietta scientists modified their data acquisition programs as required and interactively drove the vehicle several times through the region to gather data. Since then Martin Marietta personnel have gathered two additional sets of data from the red-rock area for us. We are using these data as part of a demonstration in which we bring together several techniques produced by our longer-term investigations. In the subsections below we describe both the long-term research and its application to the demonstration task of navigating through the red-rock area

Model Instantiation

The goal of the model instantiation stage is to compile as complete a model of the environment as possible prior to the definition or start of a mission. Given a specific mission, the user interactively adds mission-specific information to the environmental model. In our modeling of the red-rock region, we started with ETL's 30-meter and 5-meter digital terrain maps of the area, computed a 0.3-meter terrain model of the smaller red-rock region, and then added models of prominent discrete terrain features, such as trees and rocks. The low-resolution terrain maps provided the global context. The high-resolution map supplied a detailed ground model and key parameters for the specific object models.

We constructed the high-resolution elevation map of the red-rock region by applying a stereo technique developed by our group at SRI [Barnard88,89]. The resulting map provides a height estimate for each pixel in the original aerial images. For our images, a pixel corresponds to approximately one square foot on the ground. Although this map contains some errors, the majority of the heights are reasonable and at a resolution much higher than is available from any other source. The prominent rocks and trees are plainly visible.

We used the detailed height map to construct our initial model of the red-rock region. which consists of a terrain map and a set of labeled objects sitting on the terrain. We estimated the height of the terrain under large objects by interactively deleting them from the height map and then filling the resulting holes by interpolation. We used the Cartographic Modeling Environment [Hanson&Quam88a,b] to build three-dimensional models of the key features. which we then entered into the CKS for permanent storage.

The individual objects are represented by superquadrics with fractal textures or as faceted volumes. They are entered in the CKS according to their semantic category and location in the world. Our initial model of the red-rock region includes about ten large trees, bushes, and rocks. In the future we plan to extend our list of semantic descriptions and develop recognition techniques that are specific to these new classes.

Mission Planning

The planning system has two purposes: one is to suggest and evaluate vehicle paths for accomplishing a mission; the second is to compute and "down load" mission-specific data and instructions to the vehicle control system. A typical instruction might be to aim a sensor in a certain direction and start looking for a particular object at a specific time. So far we have concentrated on the interactive evaluation of paths and have just begun to produce data and instructions for the execution-time perception system.

Our interactive evaluation system is constructed using the CKS and the Cartographic Modeling Environment. It provides the user with the ability to generate sequences of images (i.e., movies) that correspond to the data that would be gathered by the vehicle's sensors if the vehicle followed the proposed path through a modeled world. These synthetic images provide a dramatic way to visualize a proposed path. The system can highlight key landmarks so that the user can easily determine the range of vehicle positions from which they are visible, their range of shapes, and so on. With this system we "drive through" our model of the red-rock region and select navigational landmarks for use during the execution stage.

Execution

During the execution stage, the vehicle navigates toward its destination by interpreting sensed data in terms of its predicted model of the world. To accomplish this, it performs, among other things, the following five functions: it detects unknown objects, classifies them, recognizes landmarks, tracks objects from one image to the next, and updates its world and vehicle models. Several groups, including Hughes and Carnegie-Mellon University, have demonstrated techniques for detecting unknown objects in range data. However, it is significantly harder to classify these objects into their semantic categories (e.g., rock, bush). Classification is critical for navigation because the vehicle cannot operate safely if recognition is not reliable. For example, the vehicle may be able to run cautiously over bushes, but not over rocks. We are currently investigating ways to perform this type of object classification by using the CKS to access the semantic properties of an object and the relationships between objects (see a previous section and these proceedings [Fischler&Strat89]).

Recently we have developed a strategy for incrementally updating a model of an environment during exploration by a robotic vehicle (see the next section and these proceedings [Bobick&Bolles89a]).

Recognition and Modeling of Terrain Features and Man-Made Objects

Our goal in this research effort is to develop automated methods for producing a labeled three-dimensional scene model from many images recorded from different viewpoints and from image sequences. We view the image-sequence approach as an important way to avoid many of the problems that hamper conventional stereo techniques because it provides the machine with previously unavailable information about the scene. The "redundant" information can be used to increase the precision of the data and filter out artifacts; the new information provided by the additional images can help to disambiguate matches for features that occur along occlusion boundaries and in the midst of periodic structures.

We have developed two techniques for building three-dimensional descriptions from multiple images. One is a range-based technique that builds scene models from a sequence of range images; the second is a motion analysis technique that analyzes long sequences of intensity images. The range technique uses data from an inertial guidance sensor on the vehicle to compensate for vehicle attitude and position changes caused by bumps, curves, and speed changes. As a result the range data are transformed into a static world-coordinate system, which is a necessary first step for almost all further analysis. By combining the data from multiple images, we are able to filter out artifacts and produce a more complete m p of the region in front of the vehicle. We have developed several representations of these three-dimensional data, including height maps, orientation images, and voxel arrays, each of which offers distinct coherence and resolution advantages to the analysis procedures.

Our approach to building scene models from a sequence of range images is to provide the system with a wide variety of object and terrain representations and an ability to judge the appropriateness of these representations for particular sets of data. The variety of representations is required for two reasons. First, it is needed to cover the range of object types typically found in outdoor environments. And second, it is needed to cover the range of data resolutions obtained by a robot vehicle exploring the environment.

The purpose of the evaluation procedure is to judge continually the validity of descriptions computed for objects as new data are obtained. In this scheme the model of an object typically goes through a sequence of representations as new data are gathered and processed. One of these sequences might start with a crude blob description of an initially detected object, include a detailed structural model derived from a set of high-resolution images, and end with a semantic label based on the object's description and the sensor system's task. This evolution in representations is guided by a structure we refer to as "representation space": a lattice of representations that is traversed as new information about an object becomes available. One of these representations is associated with an object only after it has been judged to be valid. We evaluate the validity of an object's description in terms of its temporal stability. We define stability in a statistical sense augmented with a set of explanations offering reasons for missing an object or having parameters change. These explanations can invoke many types of knowledge, including the physics of the sensor, the performance of the segmentation procedure, and the reliability of the matching technique. To illustrate the power of these ideas we have implemented a system, which we call TraX, that constructs and refines models of outdoor objects detected in sequences of range data gathered by an autonomous land vehicle driving cross-country [Bobick&Bolles89, *these proceedings*].

We have presented a motion analysis technique, which we call Epipolar-Plane Image (EPI) Analysis [Bolles,Baker, &Marimont87]. It is based on considering a dense sequence of images as forming a solid block of data. Slices through this solid at appropriately chosen angles intermix time and spatial data in such a way as to simplify the partitioning problem: These slices have more explicit structure than the conventional images from which they were obtained. In the paper we demonstrated the feasibility of this novel technique for building structured, three-dimensional descriptions of the world.

Recently we have extended this technique [Baker&Bolles88] to locate surfaces in the spatiotemporal solid of data, instead of analyzing slices, in order to maintain the spatial continuity of edges from one slice to the next. This surface-building process is the three-dimensional analogue of two-dimensional contour analysis. We have applied it to a wide range of data types and tasks, including medical images such as computed axial tomography (CAT) and magnetic reasonance imaging (MRI) data, visualization of higher dimensional (i.e., greater than three-dimensional) functions, modeling of objects over scale, and assessment in fracture mechanics.

We have also implemented a version of EPI analysis that works incrementally, applying a Kalman filter to update the three-dimensional description of the world each time a new image is received. As a result of these changes the program produces extended three-dimensional contours instead of sets of isolated points. These contours evolve over time. When a contour is initially detected, its location is only coarsely estimated. However, as it is tracked through several images, its shape typically changes into a smooth three-dimensional curve that accurately describes the corresponding feature in the world. We are currently planning a parallel implementation of our surface-building algorithm.

Interactive Techniques for Scene Modeling: A Cartographic Modeling Environment

Manual photointerpretation is a difficult and time-consuming step in the compilation of cartographic information. However, fully automated techniques for this purpose are currently incapable of matching the human's ability to employ background knowledge, common sense, and reasoning in the image-interpretation task. Near-term solutions to computer-based cartography must include both interactive extraction techniques and new ways of using computer technology to provide the end-user with useful information in the form of both image and map-like interactive computer displays.

In order to support research in semiautomated and automated computer-based cartography, we have developed the SRI Cartographic Modeling Environment. In the context of an interactive workstation-based system, the user can manipulate multiple images; camera models; digital terrain elevation data; point, line, and area cartographic features; and a wide assortment of three-dimensional objects. Interactive capabilities include free-hand feature entry, feature editing in the context of task-based constraints, and adjustment of the scene viewpoint. Synthetic views of a scene from arbitrary viewpoints may be constructed using terrain and feature models in combination with texture maps acquired from aerial imagery. This ability to provide an end-user with an interactively controlled scene-viewing capability could eliminate the need to produce hard-copy maps in many application contexts. Additional applications include high-resolution cartographic compilation, direct utilization of cartographic products in digital form, and generation of mission-planning and training scenarios.

Recent research has focused on developing more flexible object representations, irregular terrain grids, and improved interfaces to other systems such as the CKS. Especially important technical improvements include a reformulation of the computational model of the viewing camera to (1) reduce dynamic range problems in the Z-buffer, (2) eliminate camera transformation singularities, (3) handle scene facets that lie partially behind the camera, and (4) correctly account for perspective distortion of nearby facets in texture-mapped terrain models by recursive facet subdivision.

Our work in this area has been described in two papers, one describing basic design issues for this system [Hanson,Pentland,&Quam87], and the other providing an overview of the implementation [Hanson&Quam88].

Automated Detection and Delineation of Cultural Objects in Aerial Imagery

The detection, delineation, and recognition of any significantly broad class of objects (e.g., buildings, airports, cultivated land) in aerial imagery has proven to be an extremely difficult problem. In fact, a nominal component in the solution of this problem, image partitioning, is considered to be one of the most refractory problems in machine vision.

We have recently formulated an optimization-based approach, applicable both to image partitioning and to subsequent steps in the scene analysis process, that involves finding the "best" description of the image in terms of some specified descriptive language.

In the case of image partitioning [Leclerc89a,Leclerc89b,Leclerc89c (*these proceedings*)], we employ a language that describes the image in terms of regions having a low-order polynomial intensity variation plus white noise; region boundaries are described by a differential chain code. The best description is defined as the simplest one (in the sense of least encoding length) that is also stable (i.e., minor perturbations in the viewing conditions should not alter the description). This best description is found using a spatially local and parallel optimization algorithm (a significant improvement over the algorithm as first presented [Leclerc88]) that has been implemented on the Connection Machine. This description is further simplified where appropriate by (1) merging nonadjacent regions that can be more simply described by a single polynomial, and (2) describing the boundaries using straight lines and other more global models.

In situations where the required image description must proceed beyond that of a delineation of coherent regions, we require an extended vocabulary relevant to the semantics of the given task. Fua and Leclerc deal with the problem of boundary/shape detection given a rough estimate of where the boundary is located and a set of photometric (*intensity-gradient*) and *geometric* (shape-constraint) models for a given class of objects [Fua&Leclerc88]. They define an energy (objective) function that assumes a minimal value when the models are exactly satisfied. An initial estimate of the shape and location of the curve is used as the starting point for finding a local minimum of the energy function by embedding this curve in a viscous medium and solving the dynamic equations. This energy-minimization technique, which evolved from a less-efficient gradient-descent approach, has been implemented on the Connection Machine. It has been applied to straight-line boundary models and to more complex models that include constraints on smoothness, parallelism, and rectilinearity, and has been incorporated into the SRI Cartographic Modeling Environment described earlier. In an interactive mode, the user supplies an initial estimate of the boundary of some object (which may be quite complex, like the outline of an aeroplane) and then, if need be, corrects the optimized curve by applying forces to the curve or by changing one of a few optimized model parameters.

Automatic recognition of important cartographic objects, such as man-made structures, from aerial imagery has been addressed [Fua&Hanson89a (*these proceedings*), Fua&Hanson89b]. The basis for the approach is a theoretical formulation of object delineation as an optimization problem; practical objective measures are introduced that discriminate among a multitude of object candidates using a model language and the minimal-encoding principle. This approach is then applied in two distinct ways to the extraction of buildings from aerial imagery: the first is an operator-guided procedure that uses a massively parallel Connection Machine implementation of the objective measure [Fua89] to discover a building in real time given only a crude sketch. The second is an automated hypothesis generator that employs the objective measure during various steps in the hypothesis-generation procedure, as well as in the final stages of candidate selection; both serial and parallel (Connection Machine) approaches are implemented.

We believe that both the Leclerc and the Hanson and Fua techniques represent significant advances in the state-of-the-art in their respective areas of image partitioning and delineation of cultural features. Both systems have been able to produce excellent results in complex situations where existing (typically local) approaches fail. Future work on these techniques will emphasize the incorporation of more complex models, three-dimensional contextual information, and efficient parallel implementations.

Automated Terrain Modeling from Aerial Imagery

Stereo reconstruction is a critical task in cartography that has received a great deal of attention in the image understanding community. Its importance goes beyond the obvious application to constructing geometric models: Understanding scene geometry is necessary for effective feature extraction and other scene analysis tasks. While considerable success has been achieved in important parts of the problem, there is no complete stereo-mapping system that can perform reliably in a wide variety of scene domains.

The standard approach to the problem of stereo mapping involves finding pairs of corresponding scene points in two images (which depict the scene from different spatial locations) and using triangulation to determine scene depth. Various factors associated with viewing conditions and scene content can cause the matching process to fail; these factors include occlusion, projective or imagining distortion, featureless areas, and repeated or periodic scene structures. Some of these problems can be solved only by providing the machine with more information, which may take the form of additional images or descriptions of the global context.

Our research strategy in this task is to develop new techniques for the key steps in the stereo process, such as matching and interpolation, and, in parallel, to integrate these new ideas with existing techniques in the context of an operational system. As part of this process SRI has implemented [Hannah85] and evaluated [Hannah88] a complete high-performance stereo system. In a test of existing stereo systems on 12 pairs of digital images, conducted by the International Society of Photogrammetry, our system was able to successfully process more of the images than any other system (11 out of the 12 pairs); while no formal ranking of the test results will be published, it appears that this system placed first (or very near the top) in the competition.

We are currently investigating a number of novel approaches to stereo depth recovery that are significant departures from the conventional paradigm. In particular, Barnard has developed a hierarchical stochastic stereo-matching system, with the goal of developing practical system for cartographic analysis; the method has been implemented on SRI's Connection Machine [Barnard89a,b]. The method uses simulated annealing to find a dense disparity map that minimizes a linear combination of two functions: (1) the photometric error associated with the map, and (2) the first-order variation of the map. In other words, it finds the (approximately) smoothest map that reasonably explains the data. The method incorporates several innovative features that permit it to solve large problems in a general optimization framework — simulated annealing — that is generally considered to be very time consuming:

- It operates over a series of increasingly finer resolutions separated by one-octave low-pass or bandpass filters. By finding an approximate solution (i.e., a coarsely quantized disparity map) relatively quickly at low resolution, the system positions itself near the optimal region of its state-space at the next-higher resolution.

- It uses a new form of simulated annealing based on simulating the microcanonical ensemble. As a result, the method can be implemented efficiently with low-precision integer arithmetic and with low-quality random numbers. These are important considerations when implementing the method on a fine-grained parallel processing system.

- Because the basic simulated annealing algorithm is not readily vectorized, it cannot be implemented efficiently on a conventional supercomputer. The Connection Machine, however, is a nearly ideal architecture for this class of methods.

In parallel with the above work, Barnard is investigating a neural-network model for stereo matching that uses a continuous Hopfield-style network to minimize the same cost function used in the stochastic model [Barnard89a]. The neural network uses a representation that is even more highly parallel than the stochastic model: its representation is not a single-valued disparity (i.e., depth) map, but rather a three-dimensional lattice of disparities that is isomorphic to a sample of the three-dimensional space seen by the observer. This representation can potentially allow the method to deal with situations that are troublesome in the single-valued representation, such as occlusion, transparency, and crossover. If successful, the neural-network approach could lead to extremely fast processing through analog VLSI implementation.

While the stereo problem will remain a focus of a portion of our research, our primary effort now is to develop an understanding of how knowledge of scene depth information can be effectively used in the scene-partitioning and object-recognition tasks.

Acknowledgment

The following researchers have contributed to the work described in this report: H.H. Baker, S.T. Barnard, A. Bobick, R.C. Bolles, O. Firschein, M.A. Fischler, P. Fua, M.J. Hannah, A.J. Hanson, K.I. Laws, Y. Leclerc, L.H. Quam, G.B. Smith, T.M. Strat, and H.C. Wolf.

References

The following recent publications result fully or in part from SRI's image-understanding research program.

1. Baker, H.H., "Building Surfaces of Evolution: The Weaving Wall," *Proc. DARPA Image Understanding Workshop*, Washington, D.C., April 1988.

2. Baker, H.H., "Reimplementation of the Stanford Stereo System: Integration Experiments with the SRI Baseline Stereo System," Tech. Note 431, Artificial Intelligence Center, SRI International, Menlo Park, California, February 1989.

3. Baker, H.H., "Surface Reconstruction from Image Sequences," invited presentation, *11th European Conference on Visual Perception*, Bristol, U.K., August 1988.

4. Baker, H.H., "Generalizing Epipolar-Plane Image Analysis on the Spatiotemporal Surface," *IEEE Conference on Computer Vision and Pattern Recognition*, Ann Arbor, Michigan, June 1988.

5. Baker, H.H. and R.C. Bolles, "Generalizing Epipolar-Plane Image Analysis on the Spatiotemporal Surface," *Proc. DARPA Image Understanding Workshop*, Washington, D.C., April 1988.

6. Baker, H.H., R.C. Bolles, and D.H. Marimont, "Generalizing Epipolar-Plane Image Analysis for Non-Orthogonal and Varying View Directions," *Proc. DARPA Image Understanding Workshop*, University of Southern California, Vol.II, pp.843-848, February 1987.

7. Barnard, S., "Stochastic Stereo Matching on the Connection Machine," in *Proc. of Fourth International Conference on Supercomputing*, Santa Clara, California, 30 April - 5 May 1989.

8. Barnard, S., "Stochastic Stereo Matching on the Connection Machine," *these proceedings.*

9. Barnard, S. "Stochastic Stereo Matching Over Scale," *Int. J. of Computer Vision*, 3(1)1989.

10. Barnard, S. "Stochastic Stereo Matching Over Scale," *Proc. DARPA Image Understanding Workshop*, Washington, D.C., April 1988.

11. Barnard, S.T., "Stereo Matching by Hierarchical, Microcanonical Annealing," *Proc. DARPA Image Understanding Workshop*, University of Southern California, Vol.II, pp. 792-796, February 1987.

12. Barnard, S.T., R.C. Bolles, D. Marimont, and A.P. Pentland, "Multiple Representations for Mobile Robot Vision," *Proc. SPIE Symposium on Advances in Intelligent Robotics Systems*, Cambridge, Massachusetts, October 26-31, 1986.

13. Barnard, S.T., "A Stochastic Approach to Stereo Vision," *Proc. 5th National Conf. on Artificial Intelligence*, Philadelphia, Pennsylvania, pp.676–680, August 11-15, 1986.

14. Bobick, A.F. and R.C. Bolles, " Representation Space: An Approach to the Integration of Visual Information," *these proceedings.*

15. Bobick, A.F. and R.C. Bolles, "Representation Space: An Approach to the Integration of Visual Information," to appear in the proceedings of CVPR, San Diego, California, June 1989.

16. Bolles, R.C. and A.F. Bobick, "Exploiting Temporal Coherence in Scene Analysis for Autonomous Navigation," *Proc. Robotics and Automation Conference*, Scottsdale, Arizona, May 1989.

17. Bolles, R.C., H.H. Baker, and D.H. Marimont, "Epipolar-Plane Image Analysis: An Approach to Determining Structure from Motion," *International Journal of Computer Vision*, Kluwer Academic Publishers, Vol.I, No.1, pp. 7-55, June 1987.

18. Bolles, R.C. and H.H. Baker, "Epipolar Image Analysis: A Technique for Analyzing Motion Sequences," *Proc. 3rd. Int. Symp. on Robotics Research*, Paris, France, October 1985.

19. Corby, N.R., Mundy, J.L.,Vrobel, P.A., Hanson, A.J., Quam, L.H., Smith, G.B., and T.M. Strat, "PACE — An Environment for Intelligence Analysis," *Proc. DARPA Image Understanding Workshop*, Boston, Massachusetts, pp. 342-350 (April 6-8, 1988).

20. Firschein, O. and M.A. Fischler, "AI Application of Supercomputers: The Vision Problem," *Fourth International Conference on Supercomputing*, Santa Clara, California, April 30-May 5, 1989.

21. Fischler, M.A. and T.M. Strat, "Recognizing Objects in a Natural Environment: A Contextual Vision System (CVS)" *these proceedings.*

22. Fischler, M.A. and R.C. Bolles, "Image Understanding Research at SRI International," *Proc. DARPA Image Understanding Workshop*, University of Southern California, Vol.I, pp. 12–17, February 1987.

23. Fischler, M.A. and R.C. Bolles, "Random Sample Consensus: A Paradigm for Model Fitting with Applications to Image Analysis and Automated Cartography," *Comm. ACM*, Vol.24, No. 6, pp. 381–395, June 1981.

24. Fischler, M.A. and O. Firschein, *Intelligence: the Eye, the Brain, and the Computer*, Addison-Wesley, Reading, Massachusetts, 1987.

25. Fischler, M.A. and O. Firschein, *Readings in Computer Vision*, Morgan-Kaufmann, Los Altos, California, 1987.

26. Fischler, M.A. and O. Firschein, "Parallel Guessing: A Strategy for High-Speed Computation," *Pattern Recognition*, Vol.20, No.2, pp. 257–263, 1987.

27. P. Fua, "Object Delineation as an Optimization Problem, A Connection Machine Implementation," in *Proc. Fourth International Conference on Supercomputing*, Santa Clara, California, 30 April - 5 May 1989.

28. P. Fua and Y.G. Leclerc, "Model Driven Edge Detection," *Machine Vision and Applications, in press.*

29. P. Fua and A.J. Hanson, "Objective Functions for Feature Discrimination: Theory," *Proc. DARPA Image Understanding Workshop*, April 1989.

30. P. Fua and A.J. Hanson, "Objective Functions for Feature Discrimination: Applications to Semiautomated and Automated Feature Extraction," *Proc. DARPA Image Understanding Workshop*, April 1989.

31. Fua, P.V. and A.J. Hanson, "Extracting Generic Shapes Using Model-Driven Optimization," *Proc. DARPA Image Understanding Workshop*, Washington, D.C., April 1988.

32. Fua, P. and A.J. Hanson, "Using Generic Geometric Models for Intelligent Shape Extraction," Proc. DARA Image Understanding Workshop, University of Southern California, Vol.I, pp. 227–233, February 1987.

33. Fua, P. and A.J. Hanson, "Using Generic Geometric Knowledge to Delineate Cultural Objects in Aerial Imagery," Tech. Note 378, Artificial Intelligence Center, SRI International, Menlo Park, California, March 1986.

34. Fua, P. and A.J. Hanson, "Resegmentation Using Generic Shape: Locating General Cultural Objects," *Pattern Recognition Letters*, 1986.

35. Fua, P. and A.J. Hanson, "Locating Cultural Regions in Aerial Imagery Using Geometric Cues," *Proc. DARPA Image Understanding Workshop*, Miami Beach, Florida, pp. 271–278, December 9-10, 1985.

36. Fua, P.V. and Y.G. Leclerc, "Model Driven Edge Detection," *Proc. DARPA Image Understanding Workshop*, Washington, D.C., April 1988.

37. Fua, P. and Y. Leclerc, "Finding Object Boundaries Using Guided Gradient Ascent," *Proc. DARPA Image Understanding Workshop*, University of Southern California, Vol.II, pp. 888–891, February 1987.

38. Hannah, M.J., "A System for Digital Stereo Image Matching," submitted to *Photogrammetric Engineering and Remote Sensing*.

39. Hannah, M.J., "Digital Stereo Image Matching Techniques," *International Archives of Photogrammetry and Remote Sensing*," Vol. 27-III, pp. 280-293, XVIth ISPRS Congress, Kyoto, Japan, July 1988.

40. Hannah, M.J., "Test Results from SRI's Stereo System," *Proc. DARPA Image Understanding Workshop*, Cambridge, Massachusetts, pp. 740-744, April 1988.

41. Hannah, M.J., "Evaluation of STEREOSYS vs Other Stereo Systems," Technical Note 365, Artificial Intelligence Center, SRI International, Menlo Park, California, October 1985.

42. Hannah, M.J., "The Stereo Challenge Data Base," Technical Note 366, Artificial Intelligence Center, SRI International, Menlo Park, California, October 1985.

43. Hannah, M.J., "SRI's Baseline Stereo System," *Proc. DARPA Image Understanding Workshop*, Miami Beach, Florida, pp. 149–155, December 9-10, 1985.

44. A.J. Hanson, "Hyperquadrics: Smoothly Deformable Shapes with Convex Polyhedral Bounds," *Computer Vision, Graphics and Image Processing* **44**, 191–210, 1988.

45. A.J. Hanson and L. Quam, "A Cartographic Visualization Environment," in *Proc. of the Military Computing Conference*, Anaheim, California, pp. 233–240, May 3–5, 1988.

46. Hanson, A.J. and L. Quam, "Overview of the SRI Cartographic Modeling Environment," *Proc. DARPA Image Understanding Workshop*, Washington, D.C., April 1988.

47. Hanson, A.J., A.P. Pentland, and L.H. Quam, "Design of a Prototype Interactive Cartographic Display and Analysis Environment," *Proc. DARPA Image Understanding Workshop*, University of Southern California, Vol.II, pp. 475–482, February 1987.

48. Heeger, D. J., "Optical Flow from Spatiotemporal Filters," *Proc. First International Conference on Computer Vision*, London, England, pp. 181-190, June 8-11, 1987.

49. Laws, K.I., "Goal-Directed Textured-Image Segmentation," *Proc. SPIE Conf. on Applications of Artificial Intelligence II*, Vol. 548, Arlington, Virginia, April 9-11, 1985.

50. Laws, K.I. "Experiments in *Navigational Road Tracking*," in S.J. Rosenschein, M.A. Fischler, and L.P. Kaelbling, "Research on Intelligent Mobile Robots," Final Technical Report, Project 7390, SRI International, Menlo Park, California, pp. 151–157, May 20, 1986.

51. Leclerc, Y.G., "Constructing Simple Stable Descriptions for Image Partitioning," *International Journal of Computer Vision*, Vol.3, No.1, 1989.

52. Leclerc, Y.G., "Segmentation via Minimal-Length Encoding on the Connection Machine," *Fourth International Conference on Supercomputing*, Santa Clara, California, April 30-May 5, 1989.

53. Leclerc, G., "Image and Boundary Segmentation via Minimal-Length Encoding on the Connection Machine," *these proceedings*

54 Leclerc, Y.G., "Constructing Simple Stable Descriptions for Image Partitioning," Proc. DARPA Image Understanding Workshop, Cambridge, Massachusetts,pp. 365-382, April, 1988.

55. Leclerc, Y., "Capturing the Local Structure of Image Discontinuities in Two Dimensions," *Proc. IEEE Conf. on Computer Vision and Pattern Recognition*, San Francisco, California, pp. 34–38, June 19-23, 1985.

56. Marimont, D.H., "Projective Duality and the Analysis of Image Sequences," *Proc. of the Workshop on Motion: Representation and Analysis*, IEEE Computer Society, pp. 7–14, Kiawah Island, South Carolina, May 1986.

57. Pentland, A.P., "Fractal-Based Description of Natural Scenes," *IEEE PAMI* Vol.6, No.6, pp. 661-674, 1984.

58. Pentland, A.P., "A New Sense for Depth of Field," it Proc. 9th Int. Joint Conf. on Artificial Intelligence, Los Angeles, California, pp. 988 994, August 18-23, 1985.

59. Pentland, A.P. (ed.), *From Pixels to Predicates*, Ablex, Norwood, New Jersey, 1985.

60. Pentland, A.P., "On Describing Complex Surfaces," *Image and Vision Computing*, Vol.3, No.4, pp. 153-162, November 1985.

61. Pentland, A.P., "Part Models," *Proc. IEEE Conf. on Computer Vision and Pattern Recognition*, Miami Beach, Florida, pp. 243 249, June 22-26, 1986.

62. Pentland, A.P "Parts: Structured Descriptions of Shape," *Proc. 5th National Conf. on Artificial Intelligence*, Philadelphia, Pennsylvania, pp. 695 701, August 11-15, 1986.

63. Pentland, A.P., "Perceptual Organization and Representation of Natural Form," *Artificial Intelligence*, 28:293-331, 1986.

64. Pentland, A.P., "Recognition by Parts," SRI Technical Note 406, December 1986.

65. Quam, L.H., "The TerrainCalc System," *Proc. DARPA Image Understanding Workshop*, Miami Beach, Florida, pp. 327 330, December 9-10, 1985.

66. Smith, G.B. and T.M. Strat, "Information Management in a Sensor-Based Autonomous System," *Proc. Image Understanding Workshop*, University of Southern California, Vol.I, pp. 170–177, February 1987.

67. Smith, G.B., "Stereo Reconstruction of Scene Depth," *Proc. Computer Vision and Pattern Recognition*, San Francisco, California, pp. 271 276, June 19-23, 1985.

68. Smith, G.B., "Stereo Integral Equation," *Proc. 5th National Conf. on Artificial Intelligence*, Phildelphia, Pennsylvania, pp. 689-694, August 11-15, 1986.

69. Strat, T.M. and G.B. Smith, "Core Knowledge System: Storage and Retrieval of Inconsistent Information," *Proc. DARPA Image Understanding Workshop*, Washington, D.C., April 1988.

70. Strat, T.M. and M.A. Fischler, "One-Eyed Stereo: A General Approach to Modeling 3-D Scene Geometry," *Proc. 9th Int. Joint Conf. on Artificial Intelligence*, Los Angeles, California, pp. 937 943, August 18-23, 1985.

71. Wesley, L.P., "Evidential Knowledge-Based Computer Vision," Optical Engineering, Vol.25, No.3, pp. 363-379, March 1986.

Image Understanding Research at Carnegie Mellon

Takeo Kanade and Steve Shafer
School of Computer Science
Carnegie Mellon University
Pittsburgh PA 15213

In the last year, image understanding research at CMU has been carried out by a team of researchers led by several faculty: Takeo Kanade, Chiun-Hong Chien, Martial Hebert, Katsushi Ikeuchi, Eric Krotkov, Steve Shafer, Chuck Thorpe, and Jon Webb. We have been joined this year by Andy Witkin. The group also includes numerous graduate students, staff, and visitors. Our research spans a variety of topics in machine vision:

- Computer Vision as a Physical Science
    - Intrinsic optical models for vision
    - Model-based inspection of metal surfaces
    - Dynamic stereo based on uncertainty modeling
- Vision Algorithms
    - Hierarchical clustering by the NIHC algorithm
    - Analysis of repetitive texture
- Geometric Modeling of Objects and Sensors for Vision
    - The VANTAGE geometric/sensor modeler
    - Automatic generation of object recognition programs
    - Framework for geometric reasoning for 3-D vision
    - Robust geometric modeling
- Parallel Architectures for Vision
    - Software and Hardware for Parallel Vision
    - The 2D Machine
- Vision for Mobile Robot Navigation
    - Building terrain descriptions from range images
    - Road following using reflectance images
    - Road following by color vision
- Mobile Robot Systems
    - Map-based navigation
    - Path planning for off-road travel
    - The Autonomous Planetary Rover

1. Computer Vision as a Physical Science

Our research on physics-based methods for computer vision addresses modeling physical phenomena for robust and reliable low-level vision. It is based on the recent body of research in this area, which has begun to conclusively demonstrate that algorithms derived from models of physical processes are far more accurate and reliable than heuristically derived algorithms.

Our research in low-level vision includes basic physical models for vision, the application of these models for visual inspection, and stereo-motion analysis for robot vehicles.

1.1. Intrinsic Optical Models for Vision

The traditional approaches for low-level machine vision involve edge detection or region grouping, which do not provide very high-quality data for visual interpretation. The shortcoming of such methods is that they are based on the property of *signal coherence* -- the premise that each object or surface has a uniform intensity or color signal in the image. This premise is violated by many natural phenomena such as shadows, highlights, and surface texture. Instead, we are developing a new approach for low-level vision in which physical models are applied to the raw image data before any other analysis is performed. We call these *intrinsic models* using the terminology of Fischler, Tenenbaum, and Barrow.

Our approach has been to apply intrinsic models to conventional imagery, i.e. without special sensors such as thermal or polarization imaging, and to attempt to provide a subsitute for traditional edge detection and region grouping. To do this, we (Klinker, Shafer, Kanade) developed a segmentation algorithm based on *model coherence*, in which pixels are grouped according to their conformity with a hypothesized instantiation of a physical model, as opposed to the traditional signal coherence methods. This has been demonstrated to give better results than traditional segmentation methods [17, 18, 19]. One conclusion from this work was that general-purpose vision demands the integration of many intrinsic models, not just one at a time (which has been the usual research paradigm). In particular, tasks such as material type classification require both spatial and spectral analysis, and both kinds of analysis depend on models of both the scene and the camera.

Intrinsic models are needed for several aspects of the scene: the illumination environment, the optical properties of materials (primarily reflection), and the imaging system itself. In the last year, most of our effort in this area has been directed toward models of the imaging system (Krumm, Novak, Willson, Shafer). One of our projects has been the development of an improved Fourier-domain model of imaging through the analysis of Moire patterns. Moire patterns are very pronounced low-spatial-frequency waves caused by interference among multiple gratings. They are used very extensively for industrial surface inspection, but they also arise quite commonly in imaging fine-grained repetitive patterns. Most work in this area has been based on a "crossed grating" model in which the camera is modeled as a pair of orthogonal gratings, but this model is not completely accurate. A better model in the literature is the "sampled grating" model, which explicitly models the integration and sampling patterns of the sensor array. We have improved on this in a new "recursive sampled grating" model that also includes the effects of video transmission and subsequent digitization. The value of this model has been shown experimentally on real cameras. Moire pattern analysis appears to be a promising approach for calibration of the spatial properties of an end-to-end (lens + camera) imaging system.

We have also been studying the automation of imaging systems for high-precision robot vision, and have developed the "Imaging Space" model for calibration of an automated imaging system [30]. The Imaging Space is the configuration space for the imaging system, in which each machine-controllable parameter is one dimension. These parameters include the position and orientation of the camera, lens parameters such as zoom and focus, and other optical parameters such as filter selection and exposure time. The state of the system is a single point in the Imaging Space. In active vision, the imaging system follows a trajectory through the Imaging Space; thus, purposeful control of the imaging system can be formulated as a constraint satisfaction problem to determine the region of the Imaging Space that can provide the necessary images for the desired task. We are now beginning to develop the corresponding methods for complete geometric and radiometric calibration of the imaging equipment in the Calibrated Imaging Laboratory.

1.2. Model-Based Inspection of Metal Surfaces

Inspecting metal surfaces is one of the most difficult tasks; yet, it is one of the most frequently required tasks in many application domains. Reflectivity of metal surfaces greatly varies even though they comes from the same process. All existing shape extraction techniques, which rely on assumed surface properties, cannot handle these variations. We have been working to develop a method for determining the shape of surfaces whose reflectance properties may vary from Lambertian to specular without prior knowledge of the relative strengths of the Lambertian and specular components of reflection.

We (Nayar, Ikeuchi, Kanade) have developed a system called "photometric sampling" [28, 29]. The object surface is illuminated using the extended light sources and is viewed from a single direction. Surface illumination using extended sources makes it possible to ensure the detection of both Lambertian and specular reflections. Multiple source directions are used to obtain an image sequence of the object. An extraction algorithm uses the set of image intensity values measured at each surface point to compute orientation as well as the relative strengths of the Lambertian and specular reflection components. We have completed a 2D version of the device. Using this device, we conducted on Lambertian surfaces, specular surfaces, and hybrid surfaces whose reflectance model is composed of both Lambertian and specular components. The result show high accuracy in measured orientations and estimated reflectance parameters.

1.3. Dynamic Stereo Based on Uncertainty Modeling

Dynamic stereo, which is the ability to compute range maps continuously from stereo image sequences, has important applications in all aspects of robot navigation and manipulation. Constructing a dynamic stereo system requires judicious engineering of mathematical models and operational sensing strategies to achieve robustness, efficiency, and generality. In particular, a careful treatment of measurement uncertainty can lead to a reliable dynamic stereo system. In our previous work in this area (Matthies, Kanade, Shafer), we explored the modeling of uncertainty and how it can be applied to the interpretation of image sequences. Currently, we are developing the mathematical and the operational techniques for building a robust and effective dynamic stereo system based on these results in the explicit modeling of uncertainty.

The central mathematical components of our approach are Bayesian estimation methods applied to random field models of the range map. The random field models allow us to represent previously estimated range information in terms of a prior probability distribution for the range map. The Bayesian estimation methods provide optimal combinations of prior information and new depth measurements when doing stereo matching to compute a new range map.

These theoretical concepts are embodied in a new technique for dynamic stereo that uses small camera motions for reliable initialization of a range map in a two-camera stereo system. The small motions provide narrow-baseline image pairs from one or both cameras of the system. For initialization of the range map, range estimates computed from these narrow-baseline image pairs are used to compute a highly reliable, more precise range map by matching in a wide-baseline image pair taken from both cameras. For robots operating in difficult-to-interpret, outdoor environments, periodic verification of a range map is also essential. This can be achieved with small camera motions in a similar manner.

In 1988, we developed the basic mathematical models and operational strategies underlying the use of small camera motion to initialize stereo range maps [23, 24, 31]. Current work is continuing the experimental evaluation of these methods. A goal for future development is to combine our previous work on motion estimation (1987: Matthies and Shafer) with the new methods, leading to an integrated visual navigation system for a mobile robot.

2. Algorithms for Vision

2.1. Hierarchical Clustering by the NIHC Algorithm

Clustering techniques continue to be one of the mainstays of low-level vision. Computer vision applications of cluster analysis, such as high-dimensional Hough transforms and multispectral classification, place stringent requirements on clustering programs. These applications involve clustering at least thousands of points, detecting natural clusters in noisy data, and finding clusters in many dimensions. Many clustering procedures developed over three decades of pattern recognition research, such as K-means and single linkage, prove effective in simple situations where the problem size is small or the clusters are compact and well-separated. However, practical experience applying these procedures to computer vision problems has underscored their limitations. Our research in cluster analysis grew out of the need to develop efficient and robust natural clustering techniques for large-scale vision systems.

In 1987 we (Wallace) began experimenting with some new concepts in hierarchical clustering. We investigated an information static, Gaussian entropy, as an objective function to measure the quality of hierarchical clusterings. Information measures are not new to cluster analysis, but Gaussian entropy proved very effective in clustering the real-valued vector data typically found in computer vision applications. We abandoned the standard agglomerative-hierarchical algorithm because it is not an effective minimization procedure for the Gaussian entropy objective function. We devised a new minimization procedure, ***numerical iterative hierarchical clustering*** (NIHC), to search through the space of hierarchical clusterings to find one that minimizes Gaussian entropy. NIHC is competitive with other hierarchical procedures in time and space requirements. Moreover, in our experiments, we compared the performance of NIHC with seven other hierarchical algorithms. In once case NIHC found natural clusters where the results of the textbook algorithms were no better than random. In no case did NIHC perform significantly worse than the other procedures. Also, the clusterings produced by NIHC satisfy a partial optimality condition that does not generally hold for the output of the agglomerative hierarchical algorithm.

Two parallel versions of the NIHC are running now on commercial machines -- a shared-memory version on a 32-processor Encore Multimax, and a distributed-memory version on a 16-processor Intel IPSC/2 Hypercube. A serial version is running on a Cray-XMP. Using the NIHC, we have developed a space-time clustering program to track time-varying clusters. This has been applied to robot road-following using ERIM range and reflectance data, to color clusters in RGBxy space (color and pixel coordinates) to measure blob motion in an image sequence, and to tracking moving obstacles in two-dimensional range data.

2.2. Analysis of Repetitive Texture

We (Hamey, Kanade) have been studying computer analysis of two-dimensional regular repetitive textures in real-world images [6, 7]. Previous efforts in this field have assumed simple grid-like repetitive structure. In contrast, we assume only *locally* simple repetitive structure. This local model of repetition leads to an algorithm that is able to analyze severely distorted repetitive textures, which occur in real-world scenes. We have demonstrated the success of this algorithm on a variety of images.

An essential part of describing repetitive textures is extracting the frequency of repetition. However, regular repetitions admit many alternate frequency descriptions. We define the *fundamental frequencies* of a repetition as the two shortest independent vectors between elements of the repetition. The fundamental frequency vectors are the most perpendicular basis vectors for the repetition, and they correspond to the relative neighborhood graph of the repetitive pattern. Our algorithm exploits these properties to extract the fundamental frequencies of repetitive textures.

It is difficult to extract repetition frequency when the element of repetition is also unknown. We propose the *dominant feature assumption* as a solution to this problem. Rather than trying to find the unknown repetitive structure of an unknown texture element, we extract features from the image and rank them according to their importance (prominence). The repetitive structure of the most prominent (dominant) features is the desired structure of the entire pattern.

We have developed a four-step algorithm for understanding repetitive texture. Our experimental results demonstrate that this algorithm successfully extracts the structure of even severely distorted repetitions in real-world images.

3. Geometric Models of Objects and Sensors for Vision

Our research in high-level vision includes geometric modeling, model-based vision, and ɔD geometric reasoning.

3.1. The VANTAGE Geometric/Sensor Modeler

Geometric modeling systems allow users to create, store, and manipulate models of three-dimensional (3-D) solid objects. These geometric modeling systems have found many applications in CAD/CAM and robotics areas. One of the interesting applications is to build a model-based vision system based on a geometric modeling system. The relevant knowledge of an object for recognition is extracted from the object model in a geometric modeling system and is then used for recognition by a vision program.

A geometric modeling system represents a three-dimensional object, while a vision program observes a two-dimensional appearance and thus requires a 2D representation of a 3D object. In addition, model-based vision systems use various diverse sensors to obtain visual information. The object appearances are determined by the *product* of an object model with a sensor detectability, which tells what features the sensor can "see". Thus, it is necessary for a geometric modeling system to represent not only an object but also a sensor detectability in order for the system to be fully used for model-based vision. Suprisingly, however, little research effort has been spent in this direction, even though some of the early effort in geometric modeling comes from vision applications. This is probably because the main purpose of geometric modeling systems is to design mechanical objects and the main concern is how to represent 3D information.

We (Robert, Balakumar, Ikeuchi, Kanade) have been developed the Vantage Lisp-based solid modeling system to meet this requirement [11, 14, 16]. In particular, we designed 2D symbolic representations and the sensor modeling module. We also stabilized Vantage throu^h extensive use for various purposes such as the automatic generation of object recognition program and 3D world representation for navigation. We have shipped Vantage to several other research laboratories.

3.2. Vision Algorithm Compiler

Historically, and even today, most successful model-based vision programs are handwritten -- relevant knowledge of objects for recognition is extracted from examples of the object, tailored for the particular environment, and coded into the program by the implementors. If this is done properly, the resulting program is effective and efficient, but it requires a long development time and many vision experts.

Automatic generation of recognition programs by compilation attempts to automate this process. In particular, it extracts from the object and sensor models those features that are useful for recognition, and the control sequence which must be applied to deal with possible variations of the object appearances.

We (Ikeuchi, Hong, Kanade, Chang, Kuno) have been working on designing a geometric compiler, which automatically generates a recognition program from an object model [3, 12, 13, 15, 21]. The key techniques in designing the compiler are: object modelling, sensor modelling, prediction of appearances, strategy generation, and program generation. For object and sensor modeling in the geometric compiler, we have been using the Vantage geometric/sensor modeler. We describe geometric and photometric properties of an object using Vantage, and we have established a method to specify sensor characteristics to Vantage. Based on the geometric and photometric properties and the sensor characteristics, Vantage predicts the object appearances from various directions under various sensors.

Our geometric compiler uses *aspects*, topologically equivalent classes of object appearances, as a basic tool for object recognition. We generate a two-step program to classify one appearance of an object into one of the possible aspects, and then to determine the precise attitude and position within that aspect. We have established a technique for the geometric compiler to group the appearances systematically into aspects and then represent them symbolically. We have also established a technique to predicted ranges of uncertainty of geometric features using the sensor model. These uncertainty ranges are added to the aspect structures.

We have designed the geometric compiler to generate the aspect classification part of the interpretation tree by performing recursive sub-divisions of possible aspects [12]. This subdivision is performed by examining uncertainty ranges of aspect features in order of the smallest computational cost for finding features and determining threshold values. We represent the resulting classification strategy as a tree structure, which we refer to as an *interpretation tree*, whose intermediate nodes correspond to classification stages of aspects and store feature kinds and values for classifications. Each leaf node contains one particular aspect component. We have also finished the compiler module for attitude determination [13]. At each leaf node, the compiler examines a model face corresponding to its aspect component to decide how to define the local coordinate system on it, and stores the method in the node. At the same time, it calculates the transformation from the local coordinate system to the body coordinate system. We have also finished designing the program conversion part, to convert recognition strategies into programs [3], and we have prepared an object library, a collection of prototypical objects based on object-oriented programming.

3.3. Framework for Geometric Reasoning for 3-D Vision

Three-dimensional object description and reasoning is critical for many applications of image understanding such as robot navigation and 3-D change detection. A system for 3-D image understanding must include geometric reasoning as a primary component, because geometric relationships among object parts are a rich source of knowledge and constraint for image analysis. Unfortunately, most work in 3-D image understanding has utilized limited solid or surface models and a fixed order of analyzing image features. Such systems cannot take advantage of the specific properties or relationships in any given image and therefore perform poorly. Our research is aimed at developing a more general framework for representing 3-D models and relationships, so that vision systems can use the specific information contained in each image to its best advantage.

We (Walker, Kanade) have been developing a system called 3DFORM [32], based on the Framekit frame language defined in Common Lisp. This system has been designed with the following properties, which differentiate it from past geometric representation systems:

- *Extensible models*: 3DFORM uses frames to model object parts and geometric relations, which allows the system to be extended easily to incorporate new features. The frames are arranged in a class hierarchy, so a new class can be defined by simply specifying the differences from existing classes.
- *Flexible control flow*: The order of computation is controlled by accessing objects' attribute values, which allows the system to perform top-down and bottom-up reasoning as needed. Active procedures attached to the frames dynamically compute values as they are needed, avoiding unnecessary computations.
- *Incremental representation and reasoning*: Objects may be specified incompletely, or by constraints on them, rather than a full complete description. Constraints may be quantified by EVERY and SOME, so that even the number of parts of an object need not be fixed in advance. As constraints on an object hypothesis are evaluated, the object becomes more completely specified. When two partially specified objects are successfully matched, the result is a single object which combines the constraints of the original two.

During the past year, we have extended the matching capabilities of the 3DFORM system. When two objects are matched, the system determines whether the two objects are compatible, and if the objects are compatible creates a new object combining the constraints of both. In this way, two partial objects can be combined to create a third object which is compatible with and more complete than each of the original two. Determining compatibility and combining constraints from multiple objects is implemented by simply adding the attributes (including relationship constraints) of one object to the other, and relying on demons in the attribute slots to signal an error if incompatible attributes are added. If the objects are compatible, the result will be a single object with all the constraints from both original objects. In addition, if a particular attribute is constrained to a single value by combining the constraints from both objects, the value will be automatically computed the next time it is required. This matching process can be used not only to combine data from multiple views of a single object, but also to combine data from multiple sensors; to match a partial description of a sensed object with a previously entered model and determine the pose of the object; or to combine hypotheses derived separately into a single, more restrictive hypothesis.

Using a generic model of a building as a flat-roofed polyhedron with rectangular walls, we have demonstrated the matching capability by reading in wire frames (three-dimensional edges and vertices) corresponding to two views of a single building, forming a building hypothesis for each view, and then matching the two buildings. The same matching paradigm could be used to match 3D geometric data derived from different sensors, or from the same sensor at different times.

3.4. Robust Geometric Modeling

Shape, position, orientation, and velocity are all geometric properties, and reasoning about these properties is an important part of vision and robotics. As much as possible, we wish to automate geometric reasoning by means of geometric programs. In order that these programs be efficient, we use rounded finite precision arithmetic. Unfortunately, when we write these programs we find that it is difficult to attain reasonable reliability and almost impossible to obtain absolute reliability: there always seems to be one more special case on which the program fails. Theoretical reasons are only no coming to light as to why geometric programs are so much more difficult to make reliable than purely numerical or purely symbolic programs, but this difficulty is very commonly experienced in practice.

We (Milenkovic, Kanade) have developed techniques for creating *robust* geometric programs: geometric programs with absolutely reliable rounded arithmetic implementations [25, 26, 27]. We have focused our research on the domain of line segments in the plane, but the techniques we have developed have broader applications. The techniques are based on two principles. First, one should use a malleable representation. Specifically, replace each line segment with a *rubber band curve* which can be modified as the computation proceeds to reconcile numeric error with the symbolic structure. Second, one should keep as much of the representation implicit as possible. In the case of rubber band curves, their exact shapes are unknown at all stages of the computation. Put together, these two principles make up the *hidden variable method* which allows us to generate correct geometric information without the use of exact arithmetic. A certain amount of error is introduced by the use of rubber band curves, but this can be made to be a small fraction of the error arising from sensor noise and other measurement errors.

4. Parallel Vision

4.1. Software and Hardware for Parallel Vision

Much of the parallel vision work at Carnegie Mellon centers around the Apply language, a specialized language for low-level computer vision. We (Webb, Hamey) developed Apply initially as a way of programming the Warp machine for an important class of image processing algorithms without using W2, the basic language for the machine. Apply made is possible to develop WEB, a library of about 100 Apply programs for all sorts of low-level computer vision operations. In turn, the existence of Apply and WEB has made Warp much easier to use, and has led to interest on the part of other parallel computing groups in implementing Apply on their architecture.

We have worked with several groups in developing Apply implementations. A group at OCE Corporation in the Netherlands developed an Apply implementation on reconfigurable arrays of transputers; the Apply compiler is supposed to generate both a transputer program and an optimal configuration of the transputers for the program. At the University of Leeds, researchers implemented Apply on the Meiko Computing Surface; this implementation of Apply was recently installed at General Electric Corporate Research and Development Labs in Schenectady. At the University of Massachusetts at Amherst, an implementation of Apply on the Image Understanding Architectures is in progress. At Carnegie Mellon, the SLAP (scan-line array processor) computer has been the target of a successful Apply port.

We have also extended the functionality of existing Apply implementations. Working with Han Wang of the University of Leeds, we extended Apply on the Warp and Sun so that it could process images of varying size with a single program, and also so that it could deal with different options for border processing.

Given the success of Apply for local low-level image processing operations, it is natural to consider whether it can be extended to allow the computation of global image operations, such as histogram and Hough transform. We have considered extended Apply to use a *divide and conquer* programming model for these operations; in a paper in this workshop we show that such a model is useful, by demonstrating how the second DARPA Image Understanding benchmark can be implemented using such an extended Apply. We also prove that extending Apply in this way is generally useful; it is the case that any *reversible* image procesing operation (which gets the same result if the image is processed top to bottom or bottom to top) can be computed in this way.

Our experience in implementing the second DARPA Image Understanding benchmark on Warp was very different from the first benchmark. Because of the software tools we have developed and the maturity of the Warp hardware and software, it required the efforts of only two people to implement the complete benchmark. With the new extended Apply, even this effort would be significantly reduced.

We have also seen the continued use of our parallel vision work in the robot vehicle work at Carnegie Mellon. The SCARF road following algorithm was implemented entirely on Warp. Warp SCARF maintains its own internal state, and does all calculations necessary to predict the road position; the result is that images are fed in at one end of the Warp array, and out come road predictions at the other end. The resulting implementation is one or two orders of magnitude faster than a comparable implementation on the Sun.

4.2. The 2D Machine

Computer vision tasks are characterized by a high demand for computation. The goal of the 2D Machine project (Chien) is to design an image understanding architecture to meet this computation requirement by taking advantage of the high processing power provided by parallel machines such as Warp and Nectar [1]. The challenge is to find the best strategies for mapping data and vision tasks onto underlying parallel architectures. In order to meet this challenge, we first identify the characteristics of different vision tasks.

Vision processing can be roughly divided into three levels including low-level image processing, mid-level feature processing, and high-level object recognition:

- *Low-Level Image Processing:* Data items are pixels, which are uniformly distributed in the image space. Operations include simple local or neighborhood operations on a large amount of data. The inherent parallelism is fine-grained at the pixel level.
- *Mid-Level Feature Processing:* Data items are 2D features such as points, lines, and regions of which the distribution in the image space are not uniform. Relations among data items are spatial relationships such as adjacency, overlapping and containment. Operations on these data items include unitary operations for computing geometric properties, and binary operations involving spatial relationships. The potential parallelism is medium-grained at either the feature level or at the level of subsets of spatially adjacent features.
- *High-Level Object Recognition:* Data items are natural or cultural objects or subparts of these objects, which are not uniformly distributed in the image space. Operations include matching between objects (and their subparts) and possible models. The amount of data items in general is small, but the number of possible models may be large. The potential parallelism is at the (recognition) task level or at the level of subsets of the solution space.

The diversity of characteristics in vision tasks dictates several criteria in the design of an image understanding architecture. In order to cover this diversity, the architecture should:

- provide at least three levels of parallelism including fine-grained pixel-level parallelism, large-grained task-level parallelism, and medium-grained subtask-level parallelism;
- provide different strategies for data partitioning, dynamic data migration, tasks allocation (scheduling) and load balancing according to the types of data items, the natural of operations, current data distribution, and task dependency;
- have suitable data structures for spatial operations and for topological and part/subpart relationships; these data structures should be suitable for data partitioning and data integration as are necessary in a distributed environment;
- provide low-latency and high bandwidth communication channels for data migration and task distribution.

To facilitate data partitioning, data migration, and data integration in a distributed environment, we have selected the frame and the quadtree as the underlying data structures. For task-level parallelism, a task may consist of a frame and a desired operation which can be sent to any idle processor for processing. However, the frame structure is not a good data structure for spatial related operations (binary operations in particular) where spatial adjacency (or locality) is important to efficient processing. The quadtree, K-D tree, and R-tree are more suitable in this case. The quadtree structure has further advantages in that its regularity facilitates easy partitioning and integration of the data.

Within the framework of our proposed image understanding architecture, we have pursued the issue of distributed quadtree processing focusing on data partitioning, load balancing and dynamic data migration [4]. The results of this initial effort have been implemented on the Nectar hardware prototype currently in operation at Carnegie Mellon [1].

5. Vision for Mobile Robot Navigation

5.1. Road-Following by Color Vision

Over the last four years, we have developed ever-more-sophisticated road-following algorithms for the NAVLAB using clustering of color data. This year, we devveloped two new algorithms: SCARF and UNSCARF.

SCARF: In 1988, we (Crisman, Thorpe) completed SCARF, our system for Supervised Classification Applied to

Road Following [5]. SCARF is the logical continuation of a long chain of road following programs that use color classification. The first implementation of SCARF in 1986 ran on Sun workstations, with 32 by 30 pixel images, in about 12 seconds per image. Later implementations of that version ran on the prototype Warp and on production Warps, with speeds as fast as one image per 4 seconds.

Over the past year and a half, we have upgraded SCARF to use, first, higher resolution images (60 by 64), and, second, two images to increase dynamic range. This slowed our runs to tens of seconds per image, even on a Warp.

Now, taking advantage of compiler upgrades for the Warp's W2 language, and doing some code restructuring, we have reimplemented SCARF on the Warp. Our processing time is now down to 2 seconds per image. We moved almost all of the code onto the Warp cells themselves. Further, we reduced the number of calls to the Warp per image from 14 (last year) to 3 (earlier this year) to 1 (now). After initialization, we pass the Warp cells each new image, and get back only the new road location. All of the system state is saved on the cells from call to call. We also have debugging versions that can extract classification information for display, but those extra Warp calls and data movement slow down the system. Current running time is 1 second of Warp time per image.

The full formulation of the probability equation used in classification includes the log of the determinant of each class. Early implementations of SCARF on the Warp have always avoided logarithms, since there is no log function in W2. On benign data, this did not cause any problems. But running with the Navlab outside on a snowy day, the system did not work correctly. In our standard test sequences, each class had approximately the same size determinant (i.e., the classes had approximately equal variance), so we could safely ignore that term. But on a snowy day, the "snow" and "road" classes each had very small variance, while the "trees + parked cars + trash barrel" class had a much larger variance. This imbalance caused improper classifications. We worked with the Warp group to include a log macro and to compile it into our W2 code. The resulting system performs no better on most of our images, but dramatically improves performance on snowy days and under similar circumstances.

The resulting system has driven the Navlab many times, along our narrow bicycle path in Schenley Park. The top speed at which we have run is one meter per second, the length of our test course (compared with 20 cm/sec last year). With the fast processing loop and the complete formulation of probabilities, the vision results are solid. While vehicle speed has always been a secondary concern of our work, we can now drive at moderate speeds on our difficult test course, and should be able to use the same system to drive at higher speeds on wider, straighter roads.

UNSCARF: One of our (Crisman, Thorpe) new road detection algorithms for this past year is UNSCARF, for UNSupervised Classification Applied to Road Following [5]. A large problem with our early road perception work was dealing with rapidly changing illumination. If the sun is covered by a cloud, the lighting is diffuse and we can follow roads with a single camera. If the sun is out, there are problems with camera dynamic range, but our methods that use two cameras work. But if the sun is quickly covered or uncovered by clouds, then colors change and shadows change and the brightness changes. If object appearance differs greatly between successive processed frames, current methods have a hard time tracking the road.

UNSCARF places less emphasis on colors and more on shapes. Instead of classifying each pixel according to statistics from previous images, it groups neighboring pixels using unsupervised clustering methods. We have found that clustering with 5 parameters (R,G,B and row,col) gives us classes that are both homogeneous in color and connected in the image. We then piece a road shape together out of those clusters, instead of from individual pixels. Evaluating candidate roads uses shape cues such as parallel edges, smooth edges, edges the right distance apart, and so forth. The combination of unsupervised classification and evaluation with shape cues makes UNSCARF tolerant of the large illumination changes that have given problems to previous systems.

FERMI: FERMI deals with public highways and roads, that have more structure and variation than our Schenley Park test site (Kluge, Thorpe) [20]. The key to handling diverse roads is explicit modeling of the colors, shapes, and features of each road type. FERMI has a representation that lists width, maximum curvature, color, surface type, location of lines, type of shoulders, presence of guard rails, type of adjacent vegetation or soil, illumination conditions (sunny or cloudy), illumination direction, and so forth. By having many simple experts, one for tracking each type of feature, we are able to follow many kinds of roads within the same control framework. None of the individual trackers (edges, lines, color discontinuities, etc.) that we explored in our early work were adequate by themselves for road following. But by incorporating many of them into a single system, and intelligently selecting which tracker to use to follow which feature, we expect FERMI to be reliable and flexible. In 1988, FERMI has been designed and partially constructed, and has driven the Navlab.

5.2. Building Terrain Descriptions from Range Images

We (Hebert, Kweon, Stentz) have made progress in building terrain descriptions from range images in three areas: terrain descriptions for cross-country navigation of the NAVLAB [8], terrain map building using feature matching [8, 9], and matching of maps for high-resolution terrain descriptions [10, 22]. A terrain description based on a polygonal mesh of regions has been successfully used together with a path planner that takes into account the geometry and kinematics of the NAVLAB. The resulting enables the NAVLAB to navigate through terrain in which a description in terms of discrete objects is not sufficient. Progress in map building using feature matching includes the use of additional features such as the road edges calculated from reflectance images, and the building of maps over longer distances. Reliable feature-based map building enabled us to start actively investigating the problem of using the map built from sensor data for the navigation of a stretch of road previously explored. Since those two approaches to map building produce relatively low resolution maps, we have developed new algorithms to produce high resolution maps, that maps for which the resolution is on the order of the resolution of the scanner. In addition, an efficient algorithm that minimizes the distance between maps by a gradient descent technique leads to a better estimate of the displacement between individual and to a more accurate composite map. Even though these algorithms are somewhat computation intensive, they represent a promising direction of research for the building of high-resolution maps.

5.3. Road Following Using Reflectance Images

We (Hebert) have also been developing methods for road following using active reflectance images from the ERIM laser scanner. Active reflectance images have two characteristics that make them attractive for road-following applications: First, they are insensitive to outside illumination, that is no shadows are cast by objects in reflectance images and the influence of the level of ambiant light on the image is minimal (in fact, any program using reflectance images would work as well under night conditions). Second, each pixel in the reflectance image is also a range pixel whose position in space can be derived from the geometry of the scanner. This allows us to compute the position of the edges of the road found in a reflectance image in the vehicle's 3-D world without any of the calibration procedures that are typical of the video-based road following algorithms.

Edge detection would be the natural way of finding road edges in grey level images. The nature of the reflectance data, however, suggests the use of a region-based technique for two reasons: First, the dynamic range of the image is low, many spurious edges that are of similar strength as the road edges will be found. Second, the intensity of the road in reflectance images is very stable because it is insensitive to shadows and changes in illumination. This is to be compared with video images in which the appearance of the road region varies significantly, thus requiring the use of multiple classes of road and non-road regions. Instead of extracting the road edges directly, a road region extractor identifies the pixels that are part of the road based on the road location and appearance predicted from a previous image. This approach to road following has been successfully used for navigation over long stretches of road using the NAVLAB [9].

6. Mobile Robot Systems

6.1. Map-Based Navigation

Research on road following and map building leads naturally to the idea of closing the loop by using a map built from previous observations to guide the navigation on a portion of the world already explored. Such a capability of map based navigation would enable us to improve the performances of the vehicle in three directions:

- Faster navigation: Perception is typically the bottleneck in autonomous mobile systems because images have to be processed as often as possible to compensate for the lack of knowledge about the world. If apriori knowledge of the environment is available from previous observations, perception is needed only to periodically check that the vehicle stays on the path prescribed by the map. The perception bottleneck is therefore reduced, thus leading to faster navigation.
- More reliable navigation: Autonomous navigation is unreliable because of the uncertainty associated with any sensor data and processing. Relying more on a map means relying less on sensor data acquired during the execution of a navigation plan. Map based navigation should therefore provide more accurate navigation.
- Simpler perception: A map can provide the expected appearance of the environment at any location. That includes the expected location of objects, and the expected position and appearance of the road. This additional knowledge allows for simpler perception processing.

Athough map-based navigation algorithms could be used with a man made map (*e.g.* from surveying), using a map built from sensor information does not make any assumptions on the amount of knowledge available to the system, thus leading to a fully autonomous system. Furthermore, it is difficult to obtain the resolution of a map built from sensor data by using surveying alone.

Our (Simon, Hebert) approach to map-based road following proceeds in three steps [9]: computation of the starting position, path planning in the map, path execution and correction. The first step is needed to avoid constraining the starting position and heading of the vehicle at the beginning of the traversal of the map to those used to initiate the map building stage. The position and heading of the vehicle with respect to the map are computed by matching the features, road edges and objects, observed in an image taken at the starting position with the features of the map that are predicted to be visible given a rough initial guess of starting position. The matching algorithm is basically the same as the one used for the map building except that in the current implementation, only road edges and discrete obstacles are used. The map features are predicted by intersecting the sensor field of view with the map.

Given the starting position, the second step is to compute a path that follows the road using the map. This step is the most straightforward in that any path planner that provides for smooth paths can be used. In the current implementation, the path is computed by dividing the center curve of the road into small segments that are approximated by a small set of arcs that are later sent to the vehicle's controller.

Once a path is computed, the vehicle is ready to follow the road based on the map. Ideally, the vehicle should be able to correctly execute the path without any perception at all. In practice, however, the vehicle will drift away from the ideal path due to wheel slippage, and the accumulation of small controller errors and numerical errors. Therefore, the position and heading of the vehicle with respect to the map must be recomputed periodically by comparing the features that are actually observed while executing the path and the features that are predicted from the map given the current estimate of the vehicle's position. The question now is how often should we make a position correction, that is take an image, extract road edges and objects, and match them with the map, in order to stay within reasonable bounds of the original path. This problem is the key to map-based navigation: If the corrections are performed too often we are back to the original road following approach and we loose the benefit of having a map. If, on the other hand, we do not perform enough corrections along the path, we may drift significantly far from the nominal path and eventually run off-road. Furthermore, the corrections should be meaningful in the sense that enough features should be present at the time of the correction to ensure that the newly computed position is indeed closer to the truth than the currently available estimate. Several strategies are possible to choose the locations at which corrections should be performed. An attractive strategy is to estimate the uncertainty on the position and heading as the vehicle moves, a new correction is requested whenever the uncertainty reaches a threshold that indicates that the vehicle is too far from its nominal path. This approach guarantees that the distance between the vehicle's path and the nominal path always lies within preset bounds. It does not, however, guarantee that the images taken at the time at which a correction is needed contain enough features of interest. Another possible approach is to make a correction whenever the map predicts that features of interest may be observed from the current position. In our case, it is important to guarantee that the corrections are performed when objects are visible, since otherwise the correction would be computed on the basis of the road edges only and would therefore be ambiguous. A correction is therefore computed whenever at least one object is predicted to be visible from a position along the path. Matching the predicted objects and road edges from the map with the observed road and objects provides an unambiguous new estimate of the vehicle's position and heading.

The result of the correction calculation is an offset $\Delta=(\Delta x,\Delta y,\Delta\theta)$ between the nominal position and heading and the actual values at the time the image is taken. This offset must be used to correct the current course of the vehicle. This is achieved by shifting the path that has been executed while the image was being processed by Δ, by replanning from the current position as given by the shifted, and by replacing the pending set of motion commands by this new path.

Experiments with the NAVLAB show that it is possible to use a map to efficiently guide the navigation of an autonomous vehicle. The main benefit is that considerably fewer images have to be processed while retraversing the map. For example, a short typical stretch would require seven images to be processed using a map, while it would require at least 25 images to navigate the same stretch at the same speed. The reason for the discrepancy is that even if the position of the road were computed perfectly from each individual image, the path planner would not have information far enough in front the vehicle to plan a stable path that is guaranteed to remain on the road. Although the same results could be obtained by using a map that is entered manually, it is important to note that the combination of map building from sensor data and map-based navigation results in a fully autonomous system that can learn its environment and use its new knowledge to navigate through it.

6.2. Path Planning for Off-Road Travel

Outdoor mobile robots are used in a wide variety of scenarios ranging from those in which the robot operates with a complete map of its environment and moves between pre-determined goal points to those where the environment is completely unknown and the robot constructs a map as it explores. Regardless of the particular scenario, all navigation systems include a basic sense-plan-drive cycle for moving the robot about. The local navigator must choose the sensing points, plan paths between them, and oversee the execution of the robot's trajectory. To perform this local navigation, we (Stentz) have developed a new local path-planning method.

A number of constraints must be satisfied. First, sensing points must be selected which enable the robot to register its position relative to the world or to map new areas. Second, placements (configurations) of the robot in the environment that will incapacitate it or render it unable to locomote must be avoided. Such configurations include those that bring the robot in contact with other objects in the environment, as has been modeled in traditional indoor robotics. Outdoor robots face other hazards as well. Configurations that cause the robot to tip over or place it in situations where it cannot propel itself forward must also be avoided. Third, kinematic constraints must be taken into consideration. Most robots are not omnidirectional. They cannot travel between two arbitrary configurations within given bounds. For example, car-like vehicles cannot translate directly sideways. Fourth, uncertainty in the robot's position must be handled. This effect ranges from random error in the robot's control to gross errors such as wheel slippage. A local planner must account for such uncertainty to avoid collisions and to guarantee goal attainment.

Traditional approaches to the problem have attempted order the constraints in a hierarchy ranging from most to least severe. The constraints at the top of the hierarchy are used to plan the robot's path while those at the bottom are used to perform local modifications to the path. The problem with this approach is twofold. First, the constraint precedence can change dynamically as a function of the robot's environment. Second, there is no guarantee that local modifications to a planned path will succeed. Thrashing between levels of the hierarchy can result. We have developed a planner that models all of the constraints in a flat-level system. A single search algorithm is used to find trajectories, thus avoiding thrashing between levels of the hierarchy.

Since the pose of a car-like mobile robot can be specified using two parameters in position and one in orientation, the search space is three-dimensional. Searching this space using a dense tesselation is prohibitively expensive; instead, a recursive space-decomposition algorithm is employed. The planner attempts to plan through large subspaces (called voxels) of the search space at time, dividing the voxels and planning at a higher resolution as needed. All constraints in the system (environmental, kinematic, goal, and uncertainty) are expressed as functional inequalities. The SUP-INF method is applied to the inequalities to determine whether the constraint is satisfied for all configurations in the voxel, no configurations, or some configurations. For a given voxel, the planner first computes the maximum amount of uncertainty in the three configuration parameters that could be generated by a trajectory passing anywhere through the voxel. The voxel is then "expanded" by this uncertainty and is tested for environmental admissibility. If all configurations are inadmissible, the voxel is removed from further consideration. If a mixture exists, planning continues through the voxel at a higher resolution.

If all configurations within are admissible, the voxel is tested for goal attainment. If all configurations are in the goal space, the planner terminates with success. If a mixture exists, the planning continues at a higher resolution. If no configurations are in the goal set, the planner determines the set of points on the faces of the voxel through which the robot can leave the voxel, taking kinematic constraints into consideration. This set of points is constructed recursively and is represented using a quadtree.

The power of this technique is derived from its ability identify large subspaces through which planning can proceed quickly, taking uncertainty and goal attainment into account. The planner avoids testing individual trajectories for kinematic soundness by propagating "bundles" of trajectories through the space in a single operation. This system was implemented and tested on the NAVLAB driving in rough terrain.

6.3. The Autonomous Planetary Rover

The Autonomous Planetary Rover project, sponsored by NASA, is closely related to our NAVLAB research. In this project, we (Krotkov, Kweon, Hebert, Balakumar, Caillas) are building a a walking vehicle called the Ambler, which uses six orthogonal legs to traverse rugged terrain that wheeled vehicles can not negotiate easily [2]. The Ambler faces many of the same problems as other mobile robots, but it must also operate in rugged environments like those on Mars, at hazardous waste sites, on ocean floors, and in mines.

Perception research in this project focuses on techniques to robustly construct multi-resolution elevation maps from range imagery. The approach is to use a variety of sensors to construct the multiple resolution terrain representations necessary for tasks including locomotion, navigation, and sample acquisition. Other research aims to develop innovative gaits for legged locomotion, and to develp a centralized task control architecture to integrate the perception, planning, and control algorithms. Currently, we are experimenting with integrated systems using two testbeds: a full-scale leg, and a wheeled mobile manipulator.

7. Bibliography

1. Arnould, E., Bitz, F.J., Cooper, E.C., Kung, H.T., Sansom, R.D., and Steenkiste, P.A. The Design of Nectar: A Network Backplane for Heterogeneous Multicomputers. Proc. 3rd Intl. Conf. on Architectural Support for Programming Languages ans Operating Systems (ASPLOS III), April, 1989. To appear..

2. Bares, J., Hebert, M., Kanade, T., Krotkov, E., Mitchell, T., Simmons, R., and Whittaker, W. "An Autonomous Rover for Exploring Mars". *Computer* (1989). To appear in special issue on Autonomous Intelligent Machines..

3. Chang, J., Ikeuchi, K., and Kanade, T. Model-Based Vision System by Object-Oriented Programming. Proc. of Intl. Symposium and Exposition on Robots, Sydney, Australia, November, 1988, pp. 295-313. Longer version available as technical report CMU-RI-TR-88-3.

4. Chien, C.H. and Kanade, T. Distributed Quadtree Processing. Submitted to the Symposium on the Design and Implementation of Large Spatial Databases.

5. Crisman, J. and Thorpe, C. Color Vision for Road Following. Proc. SPIE Conf. on Mobile Robots, November, 1988. Also appeared in Proc. SIMAP 88, University of Osaka, Japan, May 1988..

6. Hamey, L.G.C. *Computer Perception of Repetitive Textures.* Ph.D. Th., Carnegie Mellon, Computer Science Dept., 1988.

7. Hamey, L.G.C. and Kanade, T. Computer Analysis of Regular Repetitive Textures. Proc. DARPA IU Workshop, May, 1989. In these proceedings..

8. Hebert, M., Kweon, I., and Kanade, T. 3-D Vision Techniques for Autonomous Vehicles. CMU-RI-TR 88-12, Carnegie Mellon Robotics Institute, 1988.

9. Hebert, M. Building and Navigating Road Scenes Using an Active Sensor. Proc. IEEE Conf. on Robotics and Automation, May, 1989. To appear..

10. Hebert, M., Caillas, C., Krotkov, E., Kweon, I., and Kanade, T. Terrain Mapping for a Roving Planetray Explorer. Proc. IEEE Conf. on Robotics and Automation, May, 1989. To appear..

11. Ikeuchi, K. and Kanade, T. Modeling Sensor Performance for Model-Based Vision. In *Robctics Research: 4*, R. Bolles and B. Roth, Ed., MIT Press, Cambridge, 1988, pp. 255-263.

12. Ikeuchi, K. and Kanade, T. "Automatic Generation of Object Recognition Program". *Proceedings of the IEEE 76*, 8 (August 1988), 1016-1035.

13. Ikeuchi, K. and Kanade, T. Applying Sensor Models of Automatic Generation of Object Recognition Programs. Proc. of 2nd Intl. Conference on Computer Vision, Tampa, Florida, December, 1988, pp. 228-237.

14. Ikeuchi, K. and Robert, J.C. Modeling Sensor Detectability With the VANTAGE Geometric/Sensor Modeler. CMU-CS 89-120, Carnegie Mellon Computer Science Dept., 1989. In this proceedings..

15. Ikeuchi, K. and Hong, K.S. Determining Linear Shape Change: Toward Automatic Generation of Object Recognition Program. Proc. of Conf. on Computer Vision and Pattern Recognition, San Diego, California, June, 1989. Longer vision available as technical report CMU-CS-88-188..

16. Kanade, T., Balakumar, P., Robert, J.C., Hoffman, R., and Ikeuchi, K. VANTAGE: A Frame-Based Geometric Modeling System. Proc. of Intl. Symposium and Exposition of Robots, Sydney, Australia, November, 1988, pp. 1405-1420.

17. Klinker, G.J. *A Physical Approach to Color Image Understanding.* Ph.D. Th., Carnegie Mellon Computer Science Dept., May 1988.

18. Klinker, G.J., Shafer, S.A., and Kanade, T. "The Measurement of Highlights in Color Images". *Intl. J. of Computer Vision 2*, 1 (June 1988), 7-32.

19. Klinker, G.J., Shafer, S.A., and Kanade, T. A Physical Approach to Color Image Understanding. Proc. 2nd Intl. Conf. on Computer Vision, December, 1988.

20. Kluge, K. and Thorpe, C. Explicit Models for Road Following. Proc. IEEE Conf. on Robotics and Automation, 1989. To appear..

21. Kuno, Y., Ikeuchi, K., and Kanade, T. Model-Based Vision by Cooperative Processing of Evidence and Hypotheses Using Configuration Spaces. Proc. SPIE Conf. #938, Orlando, Florida, April, 1988, pp. 444-453.

22. Kweon, I., Hebert, M., and Kanade, T. Perception for Rough Terrain Navigation. Proc. SPIE Mobile Robots II, Cambridge, Massachusetts, 1988.

23. Matthies, L.H. Depth from Stereo Image Sequences. International Workshop on High Precision Navigation, Stuttgart, West Germany, May, 1988.

24. Matthies, L.H., Szeliski, R., and Kanade, T. Incremental Estimation of Dense Depth Maps from Image Sequences. Conf. on Computer Vision and Pattern Recognition, Ann Arbor, Michigan, June, 1988.

25. Milenkovic, V.J. "Verifiable Implementations of Geometric Algorithms Using Finite Precision Arithmetic". *Artificial Intelligence 37* (1988), 377-401.

26. Milenkovic, V.J. *Verifiable Implementations of Geometric Algorithms Using Finite Precision Arithmetic.* Ph.D. Th., Carnegie Mellon, Computer Science Dept., July 1988.

27. Milenkovic, V.J. Robust Geometric Computations for Vision and Robotics. Proc. DARPA IU Workshop, Palo Alto, California, May, 1989. In this proceedings..

28. Nayar, S.K., Ikeuchi, K., and Kanade, T. Shape and Reflectance frmo an Image Sequence Generated Using Extended Sources. Proc. IEEE Conf. on Robotics and Automation, 1988. In this proceedings..

29. Nayar, S.K., Ikeuchi, K., and Kanade, T. Surface Roughness: Physical and Geometric Perspectives. CMU-RI-TR 89-7, Carnegie Mellon Robotics Institute, March, 1989.

30. Shafer, S.A. Automation and Calibration for Robot Vision Systems. CMU-CS 88-147, Carnegie Mellon Computer Science Dept., May, 1988.

31. Szeliski, R. *Bayesian Modeling of Uncertainty in Low Level Vision.* Ph.D. Th., Carnegie Mellon Computer Science Dept., August 1988.

32. Walker, E.L., Herman, M., and Kanade, T. "A Framework for Representing and Reasoning About Three-Dimensional Objects for Vision". *AI Magazine 9*, 2 (Summer 1988), 47-58.

Progress in Computer Vision at the University of Massachusetts[1]

Allen R. Hanson and Edward M. Riseman
Computer Vision Research Laboratory
Dept. of Computer and Information Science
University of Massachusetts
Amherst, MA 01003

ABSTRACT

This report summarize progress in image understanding research at the University of Massachusetts over the past year. Many of the individual efforts discussed in the paper are further developed in other papers in this proceedings. The summary is organized into several areas:

I. Autonomous Vehicle Navigation
II. Motion Processing
III. Knowledge-Based Interpretation
IV. Image Understanding Architecture

The research program in computer vision at UMass has as one of its goals the integration of a diverse set of research efforts into a system that is ultimately intended to achieve real-time image interpretation in a variety of vision applications. A highlight of our recent research effort is the initial integration of a perceptually-based navigation system for our local mobile robot; this work is represented in four other papers in this proceedings.

I. AUTONOMOUS VEHICLE NAVIGATION

I.1. Mobile Robot Project

The UMass Mobile Robot project is investigating the problem of enabling a mobile automaton to navigate intelligently through indoor and outdoor environments. At the foundation of our work is the premise that higher-level vision beyond the first stages of sensory processing is needed for perceptual control of the robot. In particular the system will greatly benefit from, and in many cases require, the use of knowledge and models of objects in the environment.

This project is discussed in several papers in this proceedings: our approach to world modeling, planning, and primitive task execution are presented in Fennema et al (Fennema, Hanson et al. 1989); the world model is developed in a solid modelling package, Geometer, described in (Connolly and Weiss 1989); mechanisms for optimal 2D model matching, used to locate landmarks derived from the world model and an estimate of the robot's current position, are discussed in (Beveridge, Weiss et al. 1989b); and methods for determining the pose of the robot from the matches are developed in (Kumar 1989).

In the early phases of this research, we wish to balance generality with setting sufficient constraints on the initial research goals to be achievable. Therefore, the experiments focus on robust goal-oriented navigation through a partially-modeled, unchanging environment which does not include any unmodeled obstacles. Later experiments will soften these

[1] This work was supported in part by the Defense Advanced Research Projects Agency under contract numbers F30602-87-C-0140, DACA76-85-C-0008, and DACA76-86-C-0015, and by the National Science Foundation under grant number DCR-8500332.

constraints to deal with unmodeled or moving objects, as well as learning in partially modeled or new environments.

I.2. Implementation on a Multiprocessor

Mobile robot navigation is a computationally intensive activity. One approach to achieving real-time navigation that we are investigating involves the use of a shared-memory mu'tiprocessor (Sequent Symmetry). A C-based language called RS (Pocock) has been developed for control of a set of real-time cooperative processes, and it has been ported to the Sequent. Similarly, a group of about ten high level modules involved in the navigation system are also being ported.

The near-term goal of this effort is to reproduce the experimental capability of navigating in a modeled environment. Here, the systems issues are paramount and issues of scheduling limited resources with hard real-time constraints is our research focus. We hope to demonstrate the multiprocesscr version of mobile robot navigation by early Fall,1989.

II. MOTION PROCESSING

A major area of research in our laboratory is the analysis of sequences of images derived from a moving sensor. The environments that we are examining include indoor hallway and room scenes, as well as outdoor scenes of the UMass campus. The goals of this work include the recovery of sensor motion decomposed into its translational and rotational components, the recovery of environmental depth of key surfaces and objects, and the detection of independently moving objects and, if possible, their motion parameters.

II.1. Stereoscopic Motion Analysis and the Detection of Discontinuities

One of the most important problems in stereo and motion processing is the recovery of depth and motion boundaries. A number of algorithms for computing optic flow make a global smoothness assumption that tends to unnaturally smooth across depth and motion discontinuities, and this makes later detection of these boundaries very difficult. On the other hand, knowledge of these discontinuities is very important for the flow and disparity computations to be correct, especially at occlusion boundaries.

One approach to this problem is to integrate motion and stereo data. (Balasubramanyam 1989) uses information in both the stereo and motion sequences at two time instances to define a measure of confidence in the presence of motion and depth discontinuities. This measure can be applied early, prior to the full computation of flow and disparity fields. The general idea is to use coarse disparity and flow estimates from hierarchical correlation processes (Anandan 1989) to locate and label depth and motion discontinuities; smoothing is then inhibited across these boundaries. Discontinuities that are continuous (i.e. unbroken) in the other dimension are favored. Initial results are presented on both synthetic and real stereo-motion imagery.

II.2. Smoothness Constraints for Optic Flow and Surface Reconstruction

Snyder (Snyder 1989) has developed a theoretical analysis of smoothness constraints that is used in the computation of optic flow and surface reconstruction. It is derived from a mathematical foundation that lends insight to the heuristic justification of other smoothness constraints. Under several simple assumptions he derives the most general possible smoothness constraint, which turns out to be quadratic in the first derivatives of the flow field, and quadratic in the first and second derivatives of the grey-level image intensity function. All the best-known smoothness constraints are special cases of this general form, and the relationship to a few ar examined.

II.3. 3D Interpretation of Rotational Motion from Image Trajectories

The research o^f Sawhney and Oliensis (Sawhney and Oliensis 1989) addresses the problem of discovering the motion parameters of independently moving objects in their natural coordinate system. This paper focuses on analyzing an extended time sequence of images of an object rotating uniformly around an axis of arbitrary location and orientation. It demonstrates how the abstraction of continuous descriptions of multi-frame data can lead to the recovery of scene motion and structure. Image traces of 3D feature points are generated from image point correspondences over a sequence of frames. These traces are described by continuous curves that are obtained by fitting conic arcs to the set of points. The goal is motion-based grouping of image traces to provide constraints (that are unavailable in only a few frames) sufficient to extract the motion parameters of independently moving objects in their natural coordinate system.

II.4. A Motion Data Set from the ALV

Motion analysis has remained an extremely difficult research area. One of the difficulties has been the lack of motion data with ground truth of known accuracy. In particular, this sort of data has not been collected for robot vehicles operating under realistic conditions in outdoor environments. Thus, the proper scientific evaluation of motion algorithms intended for practical application has been impossible.

In response to this general problem, our group decided to collect a reasonably large data set from the ALV (Dutta, Manmatha et al. 1989a; Dutta, Manmatha et al. 1989b) Motion sequences of about 30 frames each were collected at five different outdoor sites with different road surfaces, including on-road, dirt-road, and off-road scenarios, Data from the video camera, laser range finder, and land navigation system were recorded simultaneously under stop-and-shoot and move-and-shoot scenarios. Ground truth data was obtained using traditional surveying methods. The data is being made available to the general community and can be obtained by communicating with Ms. Valerie Cohen at UMass (E-mail address is VCohen@CS.UMass.EDU).

III. KNOWLEDGE-BASED INTERPRETATION

III.1. New Schema system

Research in knowledge-directed vision has begun to focus on two new goals. The first is to build a control system for vision that is provably near-optimal under certain assumptions. The second is to use machine learning techniques to assume part of the role of the knowledge engineer. Work is underway on the Schema System II which is designed to achieve both of these goals.

Previous research with the Schema System (Draper, Brolio et al. 1989) centered around the realization that different interpretation techniques are needed to recognize different objects. The process of finding an automobile in an image is not the same as finding a road or a tree. The problem has always been how to combine multiple recognition techniques into a single, coherent system. The Schema System II approaches this problem by viewing object recognition as search through the space of knowledge states. The general idea is to use a compile-time analysis to trace all possible paths through knowledge space in order to find the most efficient routes.

This framework is being used for learning information that would otherwise have to be provided by a "knowledge engineer." The system may be able to learn the expected cost of each knowledge source (KS) and the likelihood of each possible KS result to eliminate the need for user-defined control strategies. The system may also be able to learn which object-specific combinations of vidence allow the presence of an object to be inferred.

eliminating the need for user supplied confidence mapping functions. The final phase will address acquiring new object descriptions by learning new knowledge sources.

III.2. Perceptual Organization

III.2.1. Perceptual Organization of Curved Lines

Image curves often correspond to the bounding contours of objects as they appear in the image and provide important structural 3D information. In most cases, however, curves do not appear as coherent events in the image and must be reconstructed from fragments obtained from low level processes. Dolan (Dolan and Weiss 1989) is developing a system which exploits principles of perceptual organization, such as proximity and good continuation, to build multi-scale symbolic descriptions of co-curving or curvilinear image structure.

Two primitive geometric descriptors of image structure are employed (straight lines and conic splines) to describe the image structures of interest: collinearities, smooth curves, inflections, corners, and cusps. In order to manage the computational complexity inherent in the organization process, the system follows the iterative linking, grouping, and replacement paradigm developed by Boldt et al (Boldt and Weiss 1987; Weiss and Boldt 1986; Weiss, Hanson et al. 1985) The image primitives from which larger structures are built are unit tangents obtained by finding zero-crossings of the Laplacian and computing the local orientation of each edge point from the local gradient. Preliminary experimental results from this system may be found in (Dolan and Weiss 1989).

III.2.2. Organizing Surface Boundaries

The ability to find sets of points or lines which belong to a single object or define a single surface is extremely important in computer vision. For example, Williams and Hanson (Williams and Hanson 1988) show that when two or more image points are approximately equidistant in depth (which is often the case when they belong to a single object whose extent in depth is small relative to its distance to the camera) then the distance to those points (and therefore the object) can be accurately and reliably recovered. Similarly, the formidable combinatorics inherent in matching 3-D models to image data can only be controlled when there is prior evidence that several points or lines belong to a single object (Beveridge, Weiss et al. 1989b; Burns 1987; Burns and Kitchen 1988). Williams is developing a system for the perceptual organization of surface boundaries which exploits Gestalt grouping principles such as proximity, similarity, good continuity, convexity and symmetry.

Our view is that the goal of perceptual organization is not to assert surface boundaries in the presence of "noise", but rather to assert surface boundaries in the presence of other potentially occluding surfaces. The process of explaining an input set of line segments as the projections of boundaries of opaque surfaces at different depths begins by the insertion of virtual lines and possible vertices corresponding to potential organizations. For each member of this augmented line set, a set of constraints (derived from the physical properties of surfaces) on the role of that line in the final interpretation are asserted.

We envision this constructive or problem posing stage as being followed by an optimization or "problem answering" stage. The optimization problem consists of a linear or quadratic objective function subject to linear constraints. Each virtual line and vertex will either be promoted to a visible surface boundary, a hidden surface boundary or be deleted. As a natural side effect, wherever possible, the sign of occlusion will be determined. This aspect of the work, if successful, is tantamount to figure-ground segregation, and has important implications for obstacle avoidance in robotics and for the enforcement of smoothness constraints for creation of dense depth maps and optical flow fields.

III.3. 2D Model Matching

An important problem in model-driven 3D interpretation is how to use approximate knowledge of the location and orientation of the camera, models of objects in the environment, and the results of low-level vision to determine the image-to-model correspondence. The approach we have taken is to separate 2D model-to-image matching from the determination of the 3D pose parameters (see Section III.4)

Beveridge (Beveridge, Weiss et al. 1989a; Beveridge, Weiss et al. 1989b) assumes that a 2D model has been supplied with rough constraints on its image position (e.g. via an approximate 3D location in a modeled environment). This substantially reduces the search space of possible model-image line correspondences. The goal here is to determine correspondences between model and data lines such that an optimized spatial fit will produce the lowest match error. The search must be carried out across the space of possible line correspondences and this involves dealing with the complexities of grouping fragmented data, and missing or erroneous lines. The rotation and translation of the model that minimizes the error in spatial fit for a given set of line correspondences is computed via a closed-form solution. Interesting experimental results are achieved on images from our mobile robot domain.

III.4. 3D Pose refinement

Kumar (Kumar 1989) has developed an optimization technique for finding the 3D camera pose given a set of correspondences between 3D model lines and 2D image lines. The 3D pose is given by the rotation and translation matrices which map the world coordinate system to the camera coordinate system. Using the output of the system described in Section III.3, these algorithms allow updating of the mobile robot position via landmark recognition..

The approach is based on the constraints developed by Liu et al (Liu, Huang et al. 1988), but differs in two significant ways. First, rotation and translation are solved for simultaneously, which makes more effective use of the constraints and is more robust in the presence of noise. Second, the nonlinear least-squares optimization algorithm used to solve for rotation and translation is adapted from Horn (Horn 1987); Horn's technique provides much better convergence properties than does Liu et al's solution method based on Euler angles (Kumar 1989).

IV. IMAGE UNDERSTANDING ARCHITECTURE

IV.1. Image Understanding Benchmark

The second DARPA vision benchmark represents an integrated vision task across a range of typical vision processing (Weems, Riseman et al. 1988) Released in March 1988, the integrated benchmark involves processing from the sensory level, through intermediate processing of symbolic tokens, to matching of data against a set of models, and finally verification of the hypothesized model in a top-down manner. A sequential solution to the benchmark was written at UMass and verified by the University of Maryland.

The benchmark was widely distributed and in October 1988, UMass hosted a workshop in Avon, Connecticut to discuss the results. Representatives were present from the Darpa IU community, as well as from groups who implemented the benchmark on different machines. All of the architectures from the Strategic Computing program that are typically considered for vision applications were involved in the exercise, including the Warp, the Connection Machine, and the IUA (the Image Understanding Architecture). In addition, the benchmark was implemented on the Sun-3 and Sun-4, the Sequent Symmetry 81, Intel iPSC-2, Aspex ASP, and the Alliant FX-80. The results of this workshop are presented in (Weems, Riseman et al. 1989) in this proceedings.

IV.2. Status of IUA Project

The Image Understanding Architecture effort focussed on three main areas: completion of the IUA prototype with Hughes Research Labs; extensions to the IUA software simulator, and implementation of the DARPA Benchmark on the IUA simulator.

At the hardware level, progress towards completion of the IUA prototype included redesign of the feedback concentrator chip (for the associative functions in the CAAPP array), resubmission of the ICAP communication chips after a failed fabrication run, redesign and resubmission of the 64 processor CAAPP test chips (fabrication was delayed over six months due to vendor problems), and the design of an I/O subsystem (by Hughes). After several alternatives were considered, a decision was made to use commercially available components to construct the system controller. The first prototype board is expected to be delivered to UMass from the Hughes Research Labs in August, 1989.

At the software level, major changes and extension were made to the IUA simulator. It was ported to the Sun-3 system, integrated with the Sun windowing environment, and its interactive graphical capabilities were substantially enhanced. A full ICAP simulator was developed and merged with the CAAPP simulator to provide two-level simulation capabilities. This was then ported to the Sequent Symmetry multiprocessor and enlarged to a full sized (512x512) CAAPP and a 4096 processor ICAP. Unfortunately, memory limitations on the Sequent limited useful simulations to a 16 processor ICAP configuration. In addition, numerous improvements were made to the simulator at the basic code level and a number of vision algorithms were coded and tested.

Finally, a major effort to program the benchmark on the IUA simulator was successfully completed and reported as part of the Avon conference discussed earlier. The IUA implementation of the benchmark ran in approximately 80 msecs on the simulator.

V. REFERENCES

Anandan, P. (1989). “A Computational Framework and an Algorithm for the Measurement of Visual Motion,” IJCV, Vol. 2(3), pp. 283-310.

Balasubramanyam, P. (1989). "Early Identification of Occlusion in Stereo-Motion Image Sequences," Proc. of DARPA Image Understanding Workshop, Palo Alto, CA.

Beveridge, J. R., R. Weiss and E. Riseman. (1989a). "Matching and Fitting Sets of 2D Line Segments to Broken and Skewed Data," Dept. of Computer and Information Science, University of Massachusetts (Amherst), TR In preparation.

Beveridge, J. R., R. Weiss and E. Riseman. (1989b). "Optimizing Two-Dimensional Model Matching," Proc. of DARPA Image Understanding Workshop, Palo Alto, CA.

Boldt, M. and R. Weiss. (1987). "Token-Based Extraction of Straight Lines," Dept. of Computer and Information Science, University of Massachusetts (Amherst), TR 87-104.

Burns, J. B. (1987). "Recognition in 2D Images of 3D Objects from Large Model Bases Using Prediction Hierarchies," Proc. of International Joint Conference on Artificial Intelligence, Milan, pp. 763-766.

Burns, J. B. and L. J. Kitchen. (1988). "Rapid Object Recognition from a Large Model Base using Prediction Hierarchies," Proc. of DARPA Image Understanding Workshop, Cambridge, MA, pp. 711-719.

Connolly, C. and R. Weiss. (1989). "Geometer: A System for Modelling and Algebraic Manipulation," Proc.

of DARPA Image Understanding Workshop, Palo Alto, CA.

Dolan, J. and R. Weiss. (1989). "Perceptual Grouping of Curved Lines," Proc. of DARPA Image Understanding Workshop, Palo Alto, CA.

Draper, B., J. Brolio, R. Collins, A. Hanson and E. Riseman. (1989). "The Schema System," IJCV, Vol. 2(3), pp. 209-250.

Dutta, R., R. Manmatha, L. Williams and E. Riseman. (1989a). "A Data Set for Quantitative Motion Analysis," Proc. of Conference on Computer Vision and Pattern Recognition, San Diego, CA.

Dutta, R., R. Manmatha, L. Williams and E. Riseman. (1989b). "A Data Set for Quantitative Motion Analysis," Proc. of DARPA Image Understanding Workshop, Palo Alto, CA.

Fennema, C., A. Hanson and E. Riseman. (1989). "Toward Autonomous Vehicle Navigation," Proc. of DARPA Image Understanding Workshop, Palo Alto, CA.

Horn, B. K. P. (1987). "Closed-form solution of absolute orientation using unit quarternions," J. Opt. Soc. A., Vol. 4, pp. 629-642.

Kumar, R. (1989) "Determination of Camera Location and Orientation," Proc. of DARPA Image Understanding Workshop, Palo Alto, CA.

Liu, Y., T. Huang and O. Faugeras. (1988). "Determination of Camera Location From 2D and 3D Line and Point Correspondences," Proc. of CVPR, Ann Arbor, MI, pp. 82-88.

Pocock, G. (1989). *The Design and Analysis of a Real-Time Computer System for Robotics.* forthcoming Ph.D. Thesis, Dept. of Computer and Information Science, University of Massachusetts.

Sawhney, H. and J. Oliensis. (1989). "Description and Interpretation of Rotational Motion from Image Trajectories," Proc. of DARPA Image Understanding Workshop, Palo Alto, CA.

Snyder, M. A. (1989). "On the Mathematical Foundations of Smoothness Constraints for the Determination of Optical Flow and for Surface Reconstruction," Proc. of DARPA Image Understanding Workshop, Palo Alto, CA.

Weems, C., E. Riseman, A. Hanson and A. Rosenfeld. (1989). "A Report on the Results of the DARPA Integrated Image Understanding Benchmark Exercise," Proc. of DARPA Image Understanding Workshop, Palo Alto, CA.

Weems, C., R. Riseman, A. Hanson and A. Rosenfeld. (1988). "An Integrated Image Understanding Benchmark: Recognition of a 2 1/2 D "Mobile"," Proc. of DARPA Image Understanding Workshop, Cambridge, MA, pp. 111-126.

Weiss, R. and M. Boldt. (1986). "Geometric Grouping Applied to Straight Lines," Proc. of IEEE Computer Society Conference on Computer Vision and Pattern Recognition, Miami, FL, pp. 489-495.

Weiss, R., A. Hanson and E. Riseman. (1985). "Geometric Grouping of Straight Lines," Proc. of DARPA Image Understanding Workshop, Miami Beach, FL.

Williams, L. R. and A. R. Hanson. (1988). "Translating Optical Flow into Token Matches and Depth from Looming," Proc. of 2nd International Conference on Computer Vision, Tarpon Springs, FL, pp. 441-448.

MIT PROGRESS IN UNDERSTANDING IMAGES

T. Poggio and the staff
Artificial Intelligence Laboratory, MIT, Cambridge, MA 02139

ABSTRACT

Our program in Image Understanding has continued to focus on three main projects - the Vision Machine, object recognition and autonomous navigation -, but it has also covered a wide range of other topics from early vision modules to analog VLSI circuits for vision and theoretical work on the problem of learning. The first project - a parallel Vision Machine - has the goal of developing a system for integrating early vision modules and computing a robust description of the discontinuities of the surfaces and of their physical properties that can be used for recognition tasks. During this last year, we have interfaced the output of our integration stage with a parallel model-based recognition algorithm. The second project consists of several approaches to visual recognition. New theoretical results have been obtained and new systems have been implemented. In addition to these main themes, we have also worked on the computation and the use of motion, photogrammetry, new theoretical approaches to shape-from-shading, qualitative stereo vision, analog VLSI circuits and learning.

1 Introduction

We will briefly describe our various projects in Image Understanding. Several of them are discussed in more detail in other papers in these Proceedings and in those cases we will simply have an appropriate pointer.

2 The Vision Machine

As we described last year in a separate paper, the Vision Machine system integrates several vision cues to achieve high performance in unstructured environments, mainly for recognition tasks. It is also a tool for testing our theoretical progress in vision algorithms, their parallel implementation and their integration. The Vision Machine at presents consists of a movable two-camera Eye-Head system - the input device - and a 8K CM2. We are improving the parallel early vision algorithms which compute edge detection, stereo, motion, texture and surface color in close to real-time. The integration stage is based on the technique of coupled Markov Random Field models, and leads to a cartoon-like map of the discontinuities in the scene, with a partial labeling of the brightness edges in terms of their physical origin. In the last year, we have interfaced the output of our integration stage with a parallel model-based recognition algorithm.

2.1 The Vision Machine System

The present organization of the sys'em is shown in Figure 1. The image(s) are processed through independent algorithms or modules corresponding to different visual cues. The full integration scheme - not yet fully implemented - involves finding the various types of physical discontinuities in the surfaces - *depth discontinuities (extremal edges and blades), orientation discontinuities, specular edges, albedo edges (or marks). shadow edges* - and coupling them with each other and back to the discontinuities in the visual cues, as illustrated in Figure 1. The output of the system is a set of labeled discontinuities of the surfaces around the viewer. Thus the scheme - an instance of inverse optics - computes *surface properties*, that is attributes of the physical world and not anymore of the images. Notice that we attempt to find discontinuities in surface properties and therefore qualitative surface properties: the *inverse optics* paradigm does **not** imply that physical properties of the surfaces, such as depth or reflectance, should be extracted *precisely, everywhere*.

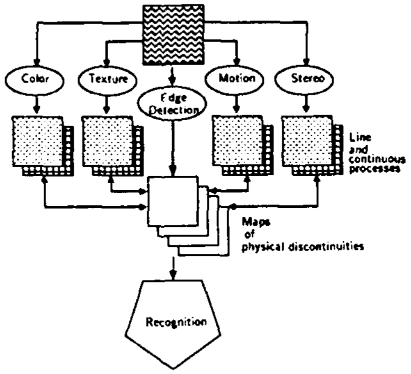

Color
Texture
Edge Detection
Motion
Stereo
Line and continuous processes
Maps of physical discontinuities
Recognition

Figure 1: Overall organization of the Vision Machine.

These discontinuities, taken together, represent a "cartoon" of the original scene which can be used for recognition and navigation (along with, if needed, interpolated depth, motion, texture and color fields). As yet we did not integrate our ongoing work on grouping in the Vision Machine. We expect to use a saliency operation on the output of the edge detection process possibly before the use of intensity edges by the MRF stage. The grouping based on T-junctions (Beymer, in preparation) should take place on the intensity edges at the same level as the MRF stage. Initial work in recognition has been integrated in the system. The Vision Machine has been demonstrated working form images to recognition through the integration of visual cues.

In the following, we will only sketch the main updates to last year's paper on the Vision Machine [43].

2.2 The Connection Machine

The new version of the Connection Machine (CM2) that we have now is a small (8K processors) configuration. The main differences with respect to the CM1 are (a) 64k bits memory per processor and (b) a floating point arithmetic accelerator, shared among 16 processors.

2.3 The Integration Stage and "deterministic" MRF

Recent work by Geiger and Girosi, based on mean-field techniques from statistical physics, has clarified the role of various parameters in the MRF technique that we use for integration and has led to interesting deterministic approximations of the stochastic procedures. These schemes have a much higher speed than the Montecarlo schemes we used so far, while promising similar performance. The work is outlined in these Proceedings. A similar deterministic sheme was developed recently by Hurlbert and Poggio to solve a specific integration problem - the integration of color with intensity edges.

2.3.1 A network for image segmentation using color

The goal of segmentation algorithms for color is to find boundaries between regions of different surface spectral reflectances and to spread uniform colors within them, without explicitly requiring the colors to be constant under changes in illumination. Hurlbert and Poggio [22] use color labels that are analogous to the CIE chromaticity coordinates x and y. Under the single source assumption, they change across space only when the surface spectral reflectance changes, except when strong specularities are present. The color edges themselves are localised with the help of luminance edges, by analogy with psychophysics of segmentation and filling-in.

Surfaces are assumed to reflect light according to the *neutral-interface-reflection* model. In this model [30] [53], the image irradiance $I(x,y,\lambda)$ is the sum of two components, the *surface* reflection and the *body* reflection:

$$I(x,y,\lambda) \;=\; L(\mathbf{r}(x,y),\lambda)[a(\mathbf{r},\lambda)g(\delta(\mathbf{r}))+bh(\delta(\mathbf{r}))],$$

where λ labels wavelength and $\mathbf{r}(x,y)$ is the point on the 3D surface to which the image coordinates (x,y) correspond. $L(\mathbf{r}(x,y),\lambda)$ is the illumination on the surface. $a(\mathbf{r},\lambda)$ is the spectral reflectance factor of the body reflection component and $g(\delta(\mathbf{r}))$ its magnitude, which depends on the viewing geometry parameters lumped together in $\delta(\mathbf{r})$. The spectral reflectance factor of the specular, or surface reflection, component b is assumed to be constant with respect to λ, as is true for inhomogeneous materials such as paints and plastics. For most materials, the magnitude of the specular component h depends strongly on the viewing geometry. Using the single source assumption, we may factor the illumination L into separate spatial and spectral components ($L(\mathbf{r},\lambda) \;=\; L(\mathbf{r})c(\lambda)$). Multiplying I by the spectral sensitivities of the color sensors $i = 1,2,3$ and integrating over wavelength yields the triplet of color values (R,G,B), where

$$R = I^R(x,y) = L(\mathbf{r}(x,y))(a^R(\mathbf{r}(x,y))g(\delta)+b^Rh(\delta))$$

and so forth and where the a^i and b^i are the reflectance factors in the spectral channels defined by the sensor spectral sensitivities.

We define the hues u and v as

$$u = \frac{R}{R+G+B}$$

and

$$v = \frac{G}{R+G+B}$$

at each pixel.

In Lambertian reflection, the specular reflectance factor b is zero. In this case, u and v are piecewise constant: they change in the image only when the $a^i(x,y)$ change. Thus u or v mark discontinuities in the surface spectral reflectance function, e.g they mark material boundaries. Conversely, image regions of constant u correspond to regions of constant surface color. Across specularities, u in general changes but often not much. Thus one approach to the segmentation problem is to find regions of "constant" u and their boundaries. The difficulty with this approach is that real u data are noisy and unreliable: u is the quotient of numbers that are not only noisy themselves but also, at least for biological photosensor spectral sensitivities, very close to one another. The goals of segmentation algorithms are therefore to enhance discontinuities in u and, within the regions marked by the discontinuities, to smoothe over the noise and fill in the data where they are unreliable. One method is to regularize – to eliminate the noise and fill in the data, while preserving the discontinuities. Using an algorithm based on Markov Random Field techniques, we have obtained encouraging results on real images [43]. The MRF technique exploits the constraint that u should be *piecewise constant* within the discontinuity contours and uses image brightness edges as guides in finding the contours. An alternative to the MRF approach is very similar to the deterministic schemes obtained by Geiger and Girosi and consists of an averaging scheme that simply replaces the value of each pixel in the hue image with the average of its local surround, iterating many times over the whole image. The algorithm takes as input the hue image (either the u-image or the v-image) and one or two edge images, either luminance edges alone, or luminance edges plus u or v edges, or u edges plus v edges. The edge images are obtained for instance by performing Canny edge detection. On each iteration, the value at each pixel in the hue image is replaced by the average of its value and those in its contributing neighborhood. A neighboring pixel is allowed to contribute if (i) it is one of the four pixels sharing a full border with the central pixel (ii) it shares the same edge label with the central pixel in all input edge images (iii) its value is non-zero and (iv) its value is within a fixed range of the central pixel value. The last requirement simply reinforces the edge label requirement when a hue image serves as an input edge image – the edge label requirement allows only those pixels that lie on the same side of an edge to be averaged, while the other insures that only those pixels with similar hues are averaged.

More formally

$$h_{i,j}^{n+1} = \frac{1}{N(C^n)} \sum_{l,m \in C(h_{i,j}^n)} h_{l,m}^n$$

where $C^n(h_{i,j}^n)$ is the set of $N(C^n)$ pixels among the next neighbors of i, j that differ from $h_{i,j}^n$ less than a specified amount and are not crossed by an edge in the edge map(s) (on the assumption that the pixel (i, j) does not belong to an edge). The iteration of this operator is similar to nonlinear diffusion and to discontinuous regularization of the type discussed by Blake and Zisserman, Geman and Geman and Marroquin [35] [4] [11]. The iterative scheme of the above equation can be derived from minimization via gradient descent of the energy function

$$E = \sum E_{i,j}$$

with

$$E_{i,j} = (1 - d_{i+1,j})V(h_{i,j}, h_{i+1,j}) + (1 - d_{i,j+1})V(h_{i,j}, h_{i,j+1})$$

$$+(1 - d_{i-1,j})V(h_{i,j}, h_{i-1,j}) + (1 - d_{i,j-1})V(h_{i,j}, h_{i,j-1}),$$

where $V(x, y) = V(x - y)$ is a quadratic potential ar[illegible] and constant for $|x - y|$ above a certain value. A simple gradient scheme finds the stationary value of h u[illegible]lving iteratively

$$dh/dt = -\alpha \frac{\partial V}{\partial h}$$

With the appropriate value of α one finds exactly the average scheme we use

$$h_{i,j}^{n+1} = < h >_C$$

where $< h >_C$ means average over the system of neighborhoods C, defined earlier and the iterations should respect the underlying system of cliques and their chromatic number [36].

Notice that our scheme is a regularization scheme that takes into account discontinuities and uses the data only as initial values for the iteration scheme. If data are taken into account in the usual way (that is equivalent to the assumption of an observation model which consists of gaussian additive noise) we have

$$h_{i,j}^{n+1} =< h >_C -k(h_{i,j}^n - h_{i,j}^0)$$

where $h^0 = u$ are the hue data. In the latter scheme, the resulting "surface" is a generalized spline. In the former scheme we expect that the algorithm will provide asymptotically piecewise constant regions bound by discontinuities.

The local averaging smoothes noise in the hue values and fills in with uniform hue regions marked by the edge inputs. On images with shading but without strong specularities, the algorithm performs a clean segmentation into regions of different hues.

2.4 Labeling the physical origin of edges: computing qualitative surface attributes

Physical Discontinuities

We classify edges according to the following physical events: discontinuities in surface properties, called *mark* or *albedo* edges (e.g., changes in the color of the surface); discontinuities in the orientation of the surface patch, called *orientation* edges (e.g., an edge in a polyhedron); discontinuities in the illumination, called *shadow* edges; *occluding boundaries*, which are discontinuities in the object space (a different object); and *specular* discontinuities, which exist for non-Lambertian objects.

Integration via Labeling with a linear classifier

Gamble, Geiger, Weinshall and Poggio have implemented a part of the general scheme. More specifically, they have used a simple linear classifier to label edges at pixels where there exists an intensity discontinuity, using the output of the line process associated with each low-level vision module. They use the fact that the modules' discontinuities are aligned, having being integrated with the intensity edges before, so that the nonexistence of a module discontinuity at a pixel is meaningful. The linear classifier corresponds to a linear network where each output unit is a weighted linear combination of its inputs (for a similar application to a problem of color vision, see [23]). The input to the network is a pixel where there exists an intensity edge and that feeds a set of qualitatively different input units. The output is a real value vector of labels' support.

In the system we have so far implemented, we achieve a rather restricted integration, since each module is integrated only with the intensity module, and labeling is done via a simple linear classifier only. It is still unclear how successfull labeling can be, using only local information.

2.5 Saliency, grouping and segmentation

A grouping and segmentation module working on the output of the edge detection module is an important part of a vision system: humans can deal with monocular, still, black and white pictures devoid of stereo, motion and color. We are now developing techniques to find salient edges, to group them and thereby segment the image. These algorithms have not been integrated yet in the Vision Machine system.

Saliency Measure

Edge maps produced by most current edge detectors are cluttered with edge responses and may have edges caused by noise. This creates difficulties for higher level processing, since the combinatorics of these algorithms often depends on the number of edge primitives being examined. What is needed is a technique to focus attention on the "important" edges in a scene. We call such attention focusing techniques that measure the "importance" of an edge saliency measures. Shimon Ullman [58] has proposed two different kinds of saliency measures: local saliency and structural saliency. An edge's local saliency is entirely determined by features of that edge alone. For example, an edge's length, its average gradient magnitude, or the color of a bounding region serve as local saliency measures. Structural saliency refers to more global properties of an edge – its relationships with other edges. Although two edges may not be locally salient, if there is a "nonaccidental" relationship between them, then the structure becomes salient. Examples of "nonaccidental" relationships, as pointed out by David Lowe, include collinearity, parallelism, and symmetry, among other things.

David Beymer has investigated local saliency measures applied to the output of the Canny edge detector. The edge features we have considered include curvature, edge length, and gradient magnitude. The measure favors those edges that have low average curvature, long length, and a high gradient magnitude. The saliency measure eliminates many of the edges due to noise and many of the unimportant edges. The edges that remain are often the long, smooth boundaries of objects and significant intensity changes inside the objects. We expect that the salient edges will help higher level processes such as grouping (structural saliency) and model based recognition by allowing them to focus attention on regions of an image bounded by salient edges.

T Junctions: Their Detection and Use in Grouping

In cluttered imagery, imagery containing many objects occluding one another, it is important to group together pieces of the image that come from the same object. In particular, given an edge map produced by the Canny edge detector, we would like to select and group together the edges from a particular object before running high level recognition algorithms on the edge data. This grouping stage helps reduce the combinatorics of the higher level stages, as they are not forced to consider false edge groupings as objects. Considering how occlusion cues can be used in grouping, we have investigated the detection of T junctions and grouping rules arising from the pairing of T junctions. When one object partially occludes another in a cluttered scene, a T junction is formed between the two objects. Beymer has developed algorithms for detecting T junctions as a postprocessing step to the Canny edge detector. The Canny edge detector, while very good at detecting edges, is particularly bad at detecting junctions. Indeed, it was designed to detect one dimensional events. This one dimensional characterization of the image breaks down at junctions since locally there are three or more surfaces in the image. We have investigated how one could use edge curvature and region properties of the image to reconstruct these "broken" junctions. Often the way Canny will fail at junctions is that one of the three curves belonging to the junction will be broken off from the other two. Beymer has modified an existing algorithm asnd achieved promisisng results in restoring broken T junctions. Once located in the image, T junctions are represented by three edges, the left part of the top horizontal edge of the T, the right part, and the stem. The top horizontal edges are the occluding edges and the vertical stem is the occluded edge. Given the junctions, we can start pairing T junctions and grouping edge fragments. If we assume that all objects in the scene fit entirely within the image boundaries, all T junctions must be matched up with a "brother" T junction along the occluded edge joining them. This constraint helps to classify T junctions, making their detection more robust. Once a T junction is matched with its brother, we know exactly which edge is the occluded edge (it is the edge that is traced to reach the brother), so we can group the two occluding edges together. The occluded edge will be extended, starting a search process to bridge the occluding object. Here we are looking for an opposing T junction on the other side of the occluding object. If such a pair of opposing Ts is found, we can group together the occluded edges of the respective T junctions. The application of these grouping rules for occluding and occluded edges often product closed contours when the Canny edges are fairly good. For each closed contour, we can form a closed region corresponding to an object or object part in the image. Finally, the T junctions are used to calculate relative depth information among the regions. In the end, the system can divide the image into regions corresponding to objects and give their relative depths. Beymer currently has the system working on "toy" images made from construction paper cutouts.

2.6 Recognition in the Vision Machine

The output of the integration stage provides a set of edges labeled in terms of physical discontinuities of the surface properties. They represent a good input to a model-based recognition algorithm like the ones described by Dan Huttenlocher and by Todd Cass in the 88 IU Proceedings. In particular, we have interfacing the Vision Machine, as implemented so far, with the Cass algorithm. We have used only discontinuities for recognition; later we will use also the information provided by the MRFs on the surface properties *between* discontinuities.

We have more ambitious goals for the recognition stage of the Vision Machine. In an unconstrained environment the library of models that a system with human-level performance requires is in the order of many thousands. Thus, the ability to learn from examples appears to be essential for the achievement of high performance in real-world recognition tasks. Learning the models becomes then a primary concern in developing a recognition system for the Vision Machine. This has not been the case in other approaches of the last few years, mainly motivated by a robotic framework, some of which will be discussed in section 3.

2.6.1 Learning in a three-stage recognition scheme

Although some of the existing recognition systems incorporate a module for learning object models from examples (e.g. Tucker's 2D system [32]) no such capability exists yet for the more difficult problems of recognizing 3D objects [28] or handwriting [8]. We believe that incorporating learning into a general-purpose recognition system may be facilitated by breaking down the task of recognition into three distinct but interacting stages: *selection*, *indexing* and *verification*.

Selection

Selection or segmentation breaks down the image into regions that are likely to correspond to single objects. The utility of an early segmentation of a scene into meaningful entities lies in the great reduction of complexity of scene interpretation. Each of the detected objects can in turn be subjected to separate recognition, by comparing it with object models stored in memory. Without prior segmentation, every possible combination of image primitives such as lines and blobs can in principle constitute an object and must be checked out. The power of early segmentation may be enhanced by integrating all available visual cues, especially if the integration parameters are automatically adjusted to suit the particular scene in question.

Indexing

By indexing we mean defining a small set of candidate objects that are likely to be present in the image. Although one cannot hope to achieve an ideal segmentation in real-world situations, partial success is sufficient if the indexing process is robust. Assuming that most objects in the real world are redundantly specified by their local features, a good indexing mechanism would use such features to overcome changes in viewpoint and illumination, occlusion and noise.

What kind of feature is good for indexing? Reliably detected lines provided by the integration of several low-level cues in the process of segmentation may suffice in many cases. We conjecture that *simple* viewpoint-invariant combinations of primitive elements, such as two lines forming a corner, parallel lines and symmetry are also likely to be useful. Ideally, only 2D information should be used for indexing, although it may be augmented sometimes by qualitative 3D cues such as relative depth.

Verification

In the verification stage each of the candidates screened by the indexing process is tested to find the best match to the image. At this stage, the system can afford to perform complicated tests, since the number of candidate objects is small. We conjecture that hierarchical indexing by a small number (two or three) features that are spatially localized in 2D suffices to achieve useful interpretations of most everyday scenes. In general, however, further verification by task-dependent routines [56] or precise shape matching, possibly involving 3D information, is required [57] [34] [28] [5] [1] [32].

As an example, T. Breuel (in these Proceedings) has considered the problem in which a recognition system is not given a fixed set of object models, but rather a set of image/label pairs from which it has to build its own object models that it subsequently uses to identify objects in new images; that is, he has considered the problem of model acquisition – in the domain of wire-frame polyedric objects – as an integral part of the problem of object recognition.

2.7 Future projects

The Vision Machine will evolve in several parallel directions:

- improvement and extensions of its early modules
- improvement of the integration and recognition stages (recognition is discussed later)
- use of the eye-head system in an active mode during recognition task by developing appropriate gaze strategies
- use of the results of the integration stage in order to improve the operation of early modules such as stereo and motion by feeding back the preliminary computation of the discontinuities

Two goals will occupy most of our attention. The first one is adevelopment of the overall organization of the Vision Machine. The system can be seen as an implementation of the *inverse optics* paradigm: it attempts to extract surface properties from the integration of image cues. It must be stressed that we never intended this framework to imply that precise surface properties such as dense, high resolution depth maps, must be delivered by the system. This extreme interpretation of inverse optics seems to be common, but was not the motivation of our project, which originally started with the name *Coarse Vision Machine* to emphasize the importance of computing qualitative, as opposed to very precise, properties of the environment. Our point of view is outlined in these Proceedings by Edelman and Poggio.

Our second main goal in the Vision machine project will be Machine Learning. In particular, we have begun to explore simple learning and estimation techniques for vision tasks. We have succeded in synthetizing a color algorithm from examples [23] and in developing a technique to perform unsupervised learning [48] of other simple vision algorithms such as simple versions of the computation of texture and stereo. In addition, we have used learning techniques to perform integration tasks, such as labeling the type of discontinuities in a scene. We have also begun to explore the connections between recent approaches to learning, such as neural networks, genetic algorithms, and classical methods in approximation theory such as splines, Bayesian techniques and Markov Random Field models. We have identified some common properties of all these approaches and some of the common limitations, such as sample complexity. As a consequence, we now believe that we can leverage our expertise in approximation techniques for the problem of learning in machine vision. Our future theoretical and computational studies will examine available learning techniques, their properties and limitations and develop new ones for the tasks of early vision, for the integration stage and for object recognition. The algorithms will be tested with the Vision Machine system and eventually incorporated into it. We will also pay attention to parallel network implementations of these algorithms: for this subgoal we will be able to leverage the work we are now doing in developing analog VLSI networks for several of the components of the Vision Machine. Towards the goal of achieving much higher flexibility in the Vision Machine we propose to explore (a) the synthesis of vision algorithms from a set of instances and (b) the refinement and tuning of preprogrammed algorithms, such as edge detection, texture discrimination, motion, color and calibration for stereo. We will also develope techniques to estimate parameters of the integration stage. Much of our effort will be focused on the new scheme for visual recognition of 3D objects, whose key component is the automatic learning of a large database of models. We aim to develop a prototype of a flexible vision system that can, in a limited way, learn from experience.

3 Object Recognition

In earlier reports, we have described a series of approaches to the problem of model-based object recognition, based on matching object shape. Our work has proceeded along a number of fronts.

3.1 Recognition from Matched Dimensionalities

Earlier reports described the work of Grimson and Lozano-Pérez on the recognition of occluded objects from noisy sensory data under the condition of matched dimensionality. Specifically, if the objects to be recognized and localized are laminar and lie on a flat surface, or if the objects are volumetric but lie in stable configurations on a flat surface, then the sensory data need only be two-dimensional (e.g. a single image); if the objects to be recognized and localized are volumetric and lie in arbitrary positions, then the sensory data must be three dimensional (e.g. stereo or motion data, laser range data). The original technique (called RAF) was designed to recognize polyhedral objects from simple measurements of the position and surface orientation of small patches of surface. The technique searches for consistent matchings between the faces of the object models and the sensory measurements, using constraints on the relative shape of pairs of model faces and pairs of measurements to reduce the search.

Our empirical work on RAF has advanced along a number of dimensions. First, we have shown that the RAF framework can successfully recognize and locate objects based on a variety of geometric features: edges, vertices, curved arcs, planar surface patches, and axes of cylinders and cones. Second, we have also shown that such features can be extracted from a range of sensory information, including grey level images, stereo data, motion data, sonar returns, laser striping data and tactile data. Third, we have shown that the RAF framework can be extended to deal with some classes of parameterized objects. These include the recognition of objects that can scale in size, the recognition of objects that are composed of rigid subparts connected through rotational degrees of freedom (e.g. a pair of scissors) and the recognition of objects that can undergo a stretching deformation along one axis.

Our empirical experience with RAF suggested that the method was remarkably efficient when dealing with data from a single object, but was inefficient when spurious data was included. To overcome this, we have incorporated a Hough transform to preselect portions of the search space on which to focus attention, and we have used thresholds on the goodness of an interpretation to terminate search. The combination of these two techniques resulted in dramatic improvement in the efficiency of the search method. Based on these observations, we have been developing a formal basis for explaining these results. In particular, we have shown the following formal results:

- If all of the data is known to have come from a single object, the expected amount of search is quadratic in the number of data and model features.

- If spurious data is included, the expected amount of search is a combination of polynomial in the number of data and model features, but exponential in the size of the actual correct interpretation.
- Using a Hough transform to preselect subspaces of the search space reduces the values of the parameters in the complexity bounds, but still leaves an exponential problem.
- Using premature termination of search based on a threshold on a "good" interpretation reduces the expected search. In particular, if the scene clutter is small enough relative to the noise in the data, the expected search becomes polynomial, otherwise it is a low order exponential.

To support the use of Hough transforms and premature termination of search, Eric Grimson and Daniel Huttenlocher have executed a formal analysis of these methods. They have derived formal characterizations for the probability of false positives in the Hough space, as a function of the noise in the data and the characteristics of the Hough transform. These results provide a means of evaluating the efficacy of the Hough transform, and suggest that one should not, in general, rely on the Hough transform to fully solve the recognition problem, but rather that one should use it as a preprocessor, selecting out small subspaces within which the RAF method can be applied effectively. The results support the empirical observations concerning the reduction in search.

Grimson and Huttenlocher have also developed a formal characterization of thresholds for terminating search, relating analytic bounds on such thresholds to expected probabilities of errors. These formal results have been shown to agree with empirical evidence from several recognition systems.

Much of our earlier work with the RAF recognition system dealt with robotics environments and the recognition of industrial parts. We have continued this effort by integrating RAF into the HANDEY task-level planning system of Lozano-Pérez. We have also continued a pilot study of applying the technique to a very different domain, underwater localization. Specifically, we have considered the problem of determining the location of an autonomous underwater vehicle by matching sensory data obtained by the vehicle against bathymetric or other maps of the environment. Sensor modalities include active methods such as sonar, and passive methods such as pressure readings and doppler data from passing ships. We have conducted some early simulation experiments using RAF, together with strategies for acquiring sensory data to solve this localization problem, with excellent results.

Our formal analysis and our empirical experience both argue that the RAF approach to recognition fails to adequately deal with the issue of segmentation of the data into subsets that are likely to have come from a single object. While the Hough transform can help reduce this problem, it is model driven, and hence potentially very expensive when applied to large libraries of objects. As an alternative to this, David Jacobs has directly addressed the issue of generic grouping in an image. Jacobs has derived measures for determining the probability that a set of edge fragments in an image is likely to have come from a single object. These measures consider simple measurements such as the separation of groups of edges, and the relative alignment of groups of edges. The recognition system, since it does not directly consider the object model, may occasionally be incorrect. However, tests of the system on a variety of images of two-dimensional and three-dimensional scenes shows a remarkable and dramatic reduction in the search required to recognize objects from a library, and also is quite effective at identifying groups of edges coming from a single object. The effect of this grouping mechanism is particularly apparent when applied to libraries of objects, since the parameters computed by the grouping scheme can be used to do effective indexing into a library.

We have also continued to investigate the use of parallel architectures, such as the Connection Machine, to obtain significant performance improvements. Todd Cass has completed the development and implementation of a parallel recognition scheme for two dimensional scenes, on which he reported in the last year Proceedings. The system uses a careful Hough transform method, followed by a sampling scheme in the parameter space to find instances of an object and its pose. Typical performance of the method involves the correct identification and localization of heavily occluded objects, in scenes in which a large number of other parts are present, in under five seconds, using a 16K processor configuration of the Connection Machine. More recent work — mentioned earlier — has focused on integrating this recognition method with data provided by the Vision Machine.

3.2 Using Recognition for Mobile Robot Localization

Besides the HANDEY task-level planning system, we have also considered the application of our recognition technology in the domain of mobile robots, especially for problems of long-term autonomy. David Braunegg has been considering the problem of constructing and maintaining a world model representation to support the navigation of long-term autonomous mobile robots. This research focuses on the use of stereo vision to obtain enough information about the world to enable path-planning and navigation-planning to be performed. Analysis of the requirements on a representation which can support long-term autonomy has led to the design of a two-level world model, an implementation of which has just been completed. This world model integrates stereo information acquired from a variety of viewpoints into a consistent whole. The system uses a variation of the RAF recognition method in conjunction with the model to recognize its location from stereo data. Experiments are currently underway to explore the efficacy of the method in integrating and maintaining noisy 3D data about an environment as the mobile robot moves within it over extended periods of time.

4 Using motion

4.1 Direct Motion Vision

Berthold Horn and some of his students are now focusing on improvements in accuracy potentially available when existing direct motion vision techniques are extended in time. Present methods recover motion and shape from instantaneous rates of change, or two adjacent frames in the discrete case. While this permits a reasonably simple formulation of the problem, the accuracy is not great, since the equivalent baseline is typically quite short. One answer to this problem would be the use of longer time intervals between frames. But then direct motion vision methods cannot be applied directly; instead, feature-based methods suited to long-range motion vision problems would need to be applied. The advantages of the direct motion vision methods, including potential for simple parallel implementation, sub-pixel accuracy in motion determination, and dense depth map recovery would be lost. One approach is to use a dynamical model of the motion imaging situation as the foundation for the systematic processing of an image sequence. In these proceedings Joe Heel describes a systematic approach to the problem of using a sequence of images to estimate a depth map and motion parameters. The approach leads to a formulation in terms of Kalman filters and to encouraging initial results with synthetic and natural images. The direct motion vision approach can also be applied to range images, where there is no concern with scale ambiguity or questions about the accuracy with which the brightness change equation is satisfied [20].

Other related work by B. Horn focuses on using additional constraint to simplify the direct motion vision problem. An important practical example of this is the use of fixation, since this introduces a constraint relating the instantaneous translation and rotation of the observer. The idea here is to have one module that tracks a designated patch in the image, while a second module recovers the motion and depth map. The tracking module uses feedback based on known least-squares algorithms for constant optical flow in a patch, as described by H.-H. Nagel and E.J. Weldon and used in the analog VLSI chip developed at Caltech by Tanner and Mead [54]. Unlike centroid-based and feature-based trackers, this method is not restricted to bright blobs against dark backgrounds or specific features such as vertices on polyhedra. All it needs is sufficient image contrast, particularly at higher spatial frequencies, and some limits on the rates of deformation of the image pattern.

4.2 Recovering 3–D Trajectories

Hildreth and Mistler are addressing the problem of recovering the 3–D trajectory of an object in space from its changing size and position in the visual image. An approximation to the instantaneous 3–D heading of an object can be derived by combining the projected image velocity of the object with a rough measure of the time-to-collision (T_c) of the object with the observer. An approximation to T_c is given by the ratio $\frac{\theta}{\dot{\theta}}$, where θ is the angular extent of the object under spherical projection, and $\dot{\theta}$ is the rate of change of this angular size over time. This approximation assumes that over short time intervals, the velocity of the object is constant, and that the object is oriented roughly parallel to the image plane. We have recently incorporated this approximation to instantaneous 3-D heading into an algorithm that derives the global parameters of motion of an object undergoing a free-fall trajectory [52]. The algorithm derives the set of parameters that yield a trajectory through space that best fits (in a least-squares sense) the time series of positions and 3-D headings of the object. Initial simulations suggest that the algorithm performs well for small objects in motion.

4.3 Structure-from-Motion with Surface Interpolation

Hildreth and Ando are exploring the integration of the recovery of 3-D structure from motion with surface interpolation. The motivations for this are two-fold. First, many of the structure-from-motion algorithms that have been proposed are feature-based, in that they first extract a set of moving features such as intensity edges, corners, points, and so on, and then derive 3-D structure at the locations of these features. If the final goal of the structure-from-motion process is to produce a complete surface representation, then restricting the initial recovery of structure to the locations of features requires a subsequent stage in which a full surface is interpolated between the depths at sparse features. Second, most algorithms also assume that a given set of features persists over an extended image sequence. It would be desirable, however, to allow the structure-from-motion process to cope with features that appear and disappear over time, for example, when occlusion and disocclusion of surfaces occurs.

We combined Ullman's incremental rigidity scheme for recovering the 3-D structure of discrete features over time with a smooth surface interpolation algorithm. As a set of features moves, its structure is built up continuously over time, and at each moment, surface interpolation is performed to fit a complete surface over the moving features. The current surface serves as an initial solution for the next moment in time, and newly appearing points in the scene are assigned an initial depth given by the interpolated surface at that image location. This scheme allows a full 3-D surface to be built up over an extended time, despite appearance and disappearance of features.

5 Photogrammetry

B. Horn is continuing to find application of known results in photogrammetry to machine vision. The optical flow equations of Prazdny and Longuett-Higgins, for example, are the parallax equations that occur in the iterative adjustment of relative orientation and exterior orientation [2]. Also, the relative orientation method using linear equations derived from eight ray pairs have been noted in photogrammetry (and discarded because of their inability to deal properly with the least-squares version of the problem) [47]. Significant further improvements of the new iterative relative orientation method reported on in last year's Image Understanding workshop resulted from a comment by a reviewer of the paper [19]. The equations simplify if the Lagrange multiplier is not eliminated and the symmetric normal equations are solved directly.

It has also been found that solutions of the relative orientation problem come in groups of four (not counting baseline reversals) if one does not consider the signs of distances along the rays (that is, whether rays intersect "in front of" or "behind" the camera). There always is at least one such group. Following a suggestion of Arun Netravali, it has been found experimentally, that while there are typically only two or three such groups of solutions, there can be four or even five. This overabundance of solutions is fortunately not a big issue, since it is rare to have positive signs on all the distances along rays on more than one of these "solutions." Also, when there is more than one solutions with all intersections in front of both imaging systems, then often one of these solutions involves very small or very large distances (often less than one hundredths or more than one hundred times the length of the baseline).

6 Shape-from-Shading

A recent revival of interest in *photoclinometry*, which is what astrogeologists call shape from shading, lead to a new look at iterative algorithms for this problem and the discovery of a novel method that can handle complex surfaces as well as integrate height and gradient information provided by other vision modules such as binocular stereo and motion vision (Horn, these Proceedings). The new method can actually obtain an exact solution if the numerical data is exact, despite the fact that it uses a regularizing term. It also lends itself to parallel implementation either on high speed digital computers or in coupled analog networks [18].

Other recent work, summarized by B. Saxberg in these Proceedings, involves the application of the dynamical systems approach to the solution of shape-from-shading problems [52]. It appears that for the first time the question of the existence of the solution may be approached. It seems now to be possible to create "impossible" shaded images—images that cannot be shaded views of any real surface under the assumed lighting and surface reflection properties. This may explain our ability to often separate albedo variations from shading effects resulting from surface shape. Also, a collection of papers on shape from shading, including a comprehensive bibliography, is about to appear [16].

7 Qualitative shape descriptors from stereo

Vision is sometimes described as a problem of inverse optics. We mentioned in a previous section that this does not require the computation of dense arrays of high-precision data of the different physical properties of surfaces. Human vision does not seem to involve the computation of the precise inverse mapping of the projection of 3D world onto a 2D retina but something more qualitative. Daphna Weinshall (these Proceedings) has concentrated on qualitative information that can be obtained from stereo disparities with little computation. More specifically, local surface patches are classified as convex, concave, hyperbolic or parabolic, using a simple function of image disparities. The axes of minimum and maximum curvature, as well as the asymptotic axes (if they exist), are also obtained. The algorithm works well with synthetic images and exact disparities. It is used to compute axes of zero curvature on real images.

8 Analog VLSI circuits for vision

As we discussed last year, our Vision Machine is mostly specialized software running on a general purpose computer, the Connection Machine. This is an ideal system for the present stage of experimentation and development. Later, it will make sense to compile the software in silicon in order to produce a faster, cheaper, and smaller Vision Machine. With funding from NSF and DuPont, we are beginning to use analog VLSI technologies to develop some initial chips as a first step toward this goal. Within the integrated circuit, the image data may be represented as a digital word or an analog value. While the advantages of digital computation are its accuracy and speed, digital circuits do not have as high functionality per device as analog circuits. Therefore, analog circuits should allow much denser computing networks. This is particularly important for the integration of computational circuitry and photosensors, which will help to alleviate the I/O bottleneck typically experienced whenever image are serially transferred between modules.

In these Proceedings Woody Yang discusses CCDs for signal processing and imaging, describing some basic operations and now those operations can be combined into a CCD processor architecture for vision. A circuit for performing Laplacian-of-Gaussian filtering of the image has been sent to fabrication. The paper discusses other CCD circuits for the integration-reconstruction stage of the Vision Machine and for stereo.

Berthold Horn has studied in an elegant way different kinds of analog networks for vision tasks [18], such as resistive grids and circuits with nonlinear components. He has focused on networks composed of locally interconnected passive elements, linear amplifiers, and simple nonlinear components [54]. While there have been excursions into the development of ideas in this area since the very beginnings of work on machine vision [14] and it was recently suggested that standard regularization algorithm can be implemented in analog networks of resistors and batteries [42], much work remains to be done. Progress will depend on careful attention to matching of the capabilities of simple networks to the needs of early vision

9 Learning

As we discussed in the last Proceedings, F. Girosi, B. Moore and T. Poggio have approached the problem of learning from the point of view of classical approximation theory. Poggio and Girosi have recently obtained what we believe is a satisfactory understanding of the learning obtained by "neural" networks such as backpropagation. In the last Proceedings we had drawn a formal analogy between simple forms of learning and hypersurface reconstruction. As a consequence, learning can be achieved by techniques such as regularization and therefore generalized splines. The connection, however, between these classical methods and feedforward networks of the backpropagation type remained unclear. Poggio and Girosi have now found that the missing link is provided by the approximation method of Radial Basis Functions. The Radial Basis Function approximation method has a sound theoretical basis and a direct interpretation in term of a feedforward network with one "hidden" layer. Poggio and Girosi have been able to prove its connections to generalized splines, to regularization techniques and to Bayes' approaches. They have developed several new extensions of the method and indicated how to address a few general issues in networks and learning within its formal framework (Girosi and Poggio, in press).

We describe briefly the interpolation and approximation technique called Radial Basis Functions (see for a review [46]), which has been used in the past for surface interpolation with very promising results [9, 13] (clearly surface reconstruction is another application of this technique of interest to vision research).

9.1 Radial Basis Functions

Given a set $D = \{(\vec{x}_i, \vec{y}_i) \in R^n \times R | i = 1, ...N\}$ of data to interpolate, the Radial Basis Function method corresponds to choosing the form of the interpolating function as

$$F(\vec{x}) = \sum_{i=1}^{N} c_i h(||\vec{x} - \vec{x}_i||^2)$$

where h is a smooth univariate function defined on $[0, \infty)$ and $||\cdot||$ is a norm on R^n. This formula means that the interpolating function is expanded on a finite N-elements basis that is given from the set of functions h translated and centered at data points. The N unknown coefficients of the expansion can be recovered imposing the interpolating conditions $F(\vec{x}_i) = Y_i$. This gives the linear system

$$Y_j = \sum_{i=1}^{N} c_i h(||\vec{x}_j - \vec{x}_i||^2) \quad j = 1, ..., N.$$

Defining the vectors $\vec{Y}$, $\vec{c}$ and the symmetric matrix H as follows

$$(\vec{Y})_i = Y_i, \qquad (\vec{c})_i = c_i, \qquad (H)_{ij} = h(||\vec{x}_j - \vec{x}_i||^2)$$

we obtain

$$\vec{c} = H^{-1}\vec{Y}$$

provided H is invertible. The invertibility of H depends on the choice of the function h. In fact Micchelli [37] proved the following theorem, that defines a class of functions that we can choose to form the basis:

Theorem 9.1.1 *Let G be a continuous function on $[0, \infty)$ and positive on $(0, \infty)$. Suppose its first derivative is completely monotonic but not constant on $(0, \infty)$. Then for any distinct vectors $\vec{x}_1, ..., \vec{x}_N \in R^n$*

$$(-1)^{n-1} det G(||\vec{x}_i - \vec{x}_j||^2) > 0$$

The interpolation conditions can be weakened if the number of knots is made lower than the number of data and their coordinates are allowed to be chosen arbitrarily [6]. In this case, denoting with $\vec{t}_1, ..., \vec{t}_K$ the coordinates of the K knots $(K < N)$ the interpolation conditions give the linear system $\vec{Y} = H\vec{c}$ where $(H)_{i\alpha} = h(||\vec{x}_i - \vec{t}_\alpha||^2)$ $(i = 1, ..., N$ and $\alpha = 1, ..., K)$. The matrix H being rectangular $(N \times K)$, this system

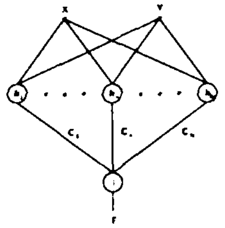


Figure 2: *The Radial Basis Function network for the interpolation of a bivariate function. The hidden unit h_n evaluates the function $h(\|\vec{x} - \vec{t}_n\|^2)$*

is overconstrained and the problem must be then regularized to obtain a reasonable set of coefficients for the expansion. A least-squares approach can then be adopted and the optimal solution can be written as

$$\vec{c} = H^+ \vec{Y}$$

where H^+ is the Moore-Penrose pseudo-inverse. In the overdetermined case, one has

$$H^+ = (H^T H)^{-1} H^T.$$

As in the previous case this formulation makes sense if the matrix $H^T H$ is non singular. Micchelli's theorem is still relevant to this problem, since Poggio and Girosi proved the following corollary:

Theorem 9.1.2 *Let G be a function satisfying the conditions of Micchelli's theorem and $\vec{x}_1, ..., \vec{x}_N$ a N-tupla of vectors in R^n. If H is the $(N - s) \times N$ matrix H obtained from the matrix $G_{i,j} = G(\|\vec{x}_i - \vec{x}_j\|^2)$ deleting s arbitrary rows, then the $(N - s) \times (N - s)$ matrix $H^T H$ is not singular.*

The first layer consists of "input" units whose number is equivalent to the number of independent variables of the problem. The second layer implements the set of radial basis function and its number of units is equal to the number of knots. The units of the second layer are in general fully connected to the units of the first one. The third layer consists of one unit (for a scalar function) connected to all the units of the second layer and computing a weighted sum of their outputs. The weights are the coefficients of the radial basis expansion and are the only unknown of the problem. Since spline interpolation can be implemented by such a network, and spline are known to have a large power of approximation we have then shown that an high degree of approximation can be obtained by just one hidden layer network.

9.2 An extension: Generalized Radial Basis Functions

Poggio and Girosi noticed that the knots of the radial basis expansion have been kept fixed, the weights being the only unknowns. To make the method more flexible they propose to consider even the knots as unknowns and to look for the configuration of weights *and* knots that minimizes the least square error on the data. The problem consists then in finding the values of the coefficients c_i and knots $\vec{t}_\alpha$ that minimizes the function

$$E = \sum_{i=1}^{N} (Y_i - \sum_{\alpha=1}^{K} c_\alpha h(\|\vec{x}_i - \vec{t}_\alpha\|^2))^2.$$

A gradient-descent approach can be adopted to find the solution to this problem. The values of c_α and $\vec{t}_\alpha$ are then regarded as the coordinates of the stable fixed point of the following dynamical system:

$$c_\alpha = -\omega \frac{\partial E}{\partial c_\alpha}, \quad \alpha = 1, ...K$$

$$\vec{t}_\alpha = -\omega \frac{\partial E}{\partial \vec{t}_\alpha}, \quad \alpha = 1, ...K$$

where ω is a parameter determining the microscopic timescale of the problem and is related to the rate of convergence to the fixed point. Defining the interpolation error as

$$\Delta_i = Y_i - \sum_{\alpha=1}^{K} c_\alpha h(\|\vec{x}_i - \vec{t}_\alpha\|^2)$$

we can write the gradient terms as

$$\frac{\partial E}{\partial c_\alpha} = -2 \sum_{i=1}^{N} \Delta_i h(\|\vec{x}_i - \vec{t}_\alpha\|^2) \quad ,$$

$$\frac{\partial E}{\partial \vec{t}_\alpha} = 4 c_\alpha \sum_{i=1}^{N} \Delta_i h'(\|\vec{x}_i - \vec{t}_\alpha\|^2)(\vec{x}_i - \vec{t}_\alpha)$$

where h' is the first derivatives of h. Equating $\frac{\partial E}{\partial \vec{t}_\alpha}$ to zero we notice that at the fixed point the knot vectors $\vec{t}_\alpha$ satisfy the following equation:

$$\vec{t}_\alpha = \frac{\sum_i P_i^\alpha \vec{x}_i}{\sum_i P_i^\alpha}$$

where $P_i^\alpha = \Delta_i h'(\|\vec{x}_i - \vec{t}_\alpha\|^2)$. The optimal knots are then a weighted sum of the data points. The weight P_i^α of the data point i for a given knot α is high if the interpolation error Δ_i is high there *and* the radial basis function centered on that knot changes quickly in a neighbor of the data point.

9.3 RBF are equivalent to regularization

Interesting connections between RBF and regularization techniques arise when the basis function are chosen to be Gaussian. Let us consider the RBF method in its original formulation, having chosen the basis function to be a Gaussian G. The coefficients of the expansion are the solution of the linear system $\vec{Y} = G\vec{c}$ where $(G)_{ij} = G(\|\vec{x}_i - \vec{x}_j\|^2)$. If data are noisy a well known technique [55] to regularize the solution is to substitute the previous linear system with the following

$$\vec{Y} = (G + \lambda I)\vec{c}$$

where λ is a small parameter and I is the identity matrix. We now show that the same approximating function can be obtained from a pure regularization approach. Let us consider the following functional

$$E_1[F] = \sum_i (Y_i - F(\vec{x}_i))^2 + \lambda \int d\vec{x} \sum_{m=0}^{\infty} a_m (D^m F(\vec{x}_i))^2$$

where λ is a parameter, $D^{2m} = \nabla^{2m}$, $D^{2m+1} = \vec{\nabla}\nabla^{2m}$, ∇^2 is the Laplacian operator and the coefficients a_m are to be chosen. It can be easily proved [61] that by posing $a_m = \frac{\sigma^{2m}}{m! 2^m}$ the function that minimizes this functional can be written as

$$F(\vec{x}) = \sum_{i=1}^{N} c_i G(\|\vec{x} - \vec{x}_i\|^2) \tag{1}$$

where G is a Gaussian of variance σ and the coefficients satisfy the linear system $\vec{Y} = (G + \lambda I)\vec{c}$, that is the same as before. So in this case RBF and regularization are equivalent. Notice that changing the coefficients a_m is equivalent to selecting another basis function h instead of G. In fact it can be shown that the set a_m and h are related by the following distributional partial differential equation:

$$\sum_{m=0}^{\infty} (-1)^m a_m \nabla^{2m} h(\vec{x}) = \delta(\vec{x}) \quad .$$

The stabilizer described above is not the most general one. Other types could have been chosen, depending on the a priori information about the surface to be reconstructed. The previous one is suitable if we want to keep local the interaction between a data point and its neighbors, since the Gaussian falls off very quikly, that is the "interaction" is short range. It can be shown that this is related to the presence of a term of degree zero in the stabilizer [61]. For example, in two dimensions, if we chose a stabilizer like

$$\int dxdy \left[\left(\frac{\partial^2 F}{\partial x^2} \right)^2 + 2 \left(\frac{\partial^2 F}{\partial x \partial y} \right)^2 + \left(\frac{\partial^2 F}{\partial y^2} \right)^2 \right]$$

this leads to a Radial Basis Function of the type $h(\|\vec{x}\|^2) = \|\vec{x}\|^2 log\|\vec{x}\|$. This kind of interaction is clearly long-range, as it should be, since the corresponding functional is the bending energy of a thin plate of infinite extent (Duchon and Meinguet gave the name *thin plate splines* to the solution of the interpolation problem obtained minimizing this functional).

The same kind of results can be obtained in a third way, in the networks framework. Let us consider the network of fig. 2 and the problem to find the "synaptic" weights. If we adopt a least square criterion we recover the usual linear system $\vec{Y} = G\vec{c}$, but often it is considered an advantage to keep the connections from growing to infinity, and so the following functional is minimized:

$$E_2[F] = \sum_i (Y_i - \sum_{i=1}^{N} c_i G(\|\vec{x} - \vec{x}_i\|^2))^2 + \lambda \sum_i c_i^2$$

where the last term gives an high price to the configurations in which some coefficient c_i is very high. It is immediate to see that the minimization of this functional leads to the solution of the linear system $\vec{Y} = (G + \lambda I)\vec{c}$. This shows the equivalence between some of the "new" neural networks techniques and classical regularization.

References

[1] N. Ayache and O. D. Faugeras. Hyper: A new approach for the recognition and positioning of two-dimensional objecs. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, PAMI-8(1):44–54, 1986.

[2] L.U. Bender. Derivation of parallax equations. *Photogrammetric Engineering*, 33(10):1175–1179, October 1967.

[3] Mario Bertero, Tomaso Poggio, and Vincent Torre. Ill-posed problems in early vision. *Proceedings of the IEEE*, 76:869–889, 1988.

[4] Andrew Blake and Andrew Zisserman. *Visual Reconstruction*. MIT Press, Cambridge, Mass, 1987.

[5] Robert C. Bolles, Patrice Horaud, and M.J. Hannah. 3dpo: A three-dimensional part orientation system. In *Proceedings IJCAI*, pages 1116–1120, 1983.

[6] D.S. Broomhead and D. Lowe. Multivariable functional interpolation interpolation and adaptive networks. *Complex Systems*, 2:321–355, 1988.

[7] Heinrich H. Bülthoff, James J. Little, and Tomaso Poggio. A parallel algorithm for real-time optical flow. *Nature*, 337:549 – 553, February 1989.

[8] S. Edelman and S. Ullman. Reading cursive script by computer, 1989. to appear in 42nd SPSE Conference Proceedings.

[9] R. Franke. Scattered data interpolation: tests of some method. *Math. Comp.*, 38(5):181–200, 1982.

[10] Davi Geiger and Tomaso Poggio. An optimal scale for edge detection. A.I. Memo No. 1078, Artificial Intelligence Laboratory, Massachusetts Institute of Technology, September 1988.

[11] S. Geman and D. Geman. Stochastic relaxation, Gibbs distributions, and the Bayesian restoration of images. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, PAMI-6:721-741, 1984.

[12] W. Eric L. Grimson and Daniel P. Huttenlocher. On the sensitivity of the hough transform for object recognition. A.I. Memo 738 1044, Artificial Intelligence Laboratory, Massachusetts Institute of Technology, May 1988.

[13] R.L. Hardy. Multiquadric equations of topography and other irregular surfaces. *J. Geophys. Res*, 76:1905–1915, 1971.

[14] Joachim Heel. Dynamical systems and motion vision. A.I. Memo No. 1037, Artificial Intelligence Laboratory, Massachusetts Institute of Technology, April 1988.

[15] Ellen C. Hildreth and Shimon Ullman. The computational study of vision. A.I. Memo No. 1038, Artificial Intelligence Laboratory, Massachusetts Institute of Technology, April 1986.

[16] Berthold K. P. Horn and M.J. Brooks. *Shape from Shading*. MIT Press, Cambridge, Mass., 1989.

[17] Berthold K.P. Horn. Determining lightness from an image. *Computer Graphics and Image Processing*, 3(1):277–299, December 1974.

[18] Berthold K.P. Horn. Parallel networks for machine vision. A.I. Memo No. 1071, Artificial Intelligence Laboratory, Massachusetts Institute of Technology, Cambridge, MA, December 1988.

[19] Berthold K.P. Horn. “relative orientation. In *Proceedings Image Understanding Workshop*, pages 181–190, Los Angeles, CA, February 1988. Morgan Kaufmann, San Mateo, CA.

[20] Berthold K.P. Horn and John Harris. Rigid body motion from range image sequences. *International Journal of Computer Vision*, 1989.

[21] Ian D. Horswill and Rodney A. Brooks. Situated vision in a dynamic world: Chasing objects. In *AAAI*, pages 796–800, St. Paul, MN, August 1988.

[22] A. Hurlbert and T. Poggio. A network for image segmentation using color. *To be published in the Proceedings of the Denver Conference on Neural Networks*, November 1988.

[23] A. Hurlbert and T. Poggio. Synthetizing a color algorithm from examples. *Science*, 239:482–485, 1988.

[24] Anya Hurlbert and Tomaso Poggio. Learning a color algorithm from examples. In *Neural Information Processing Systems: Proceedings of the Neural Information Processing Conference*, pages 622–631, New York, NY, 1988. American Institute of Physics.

[25] Anya Hurlbert and Tomaso Poggio. Making machines (and ai) see. *Daedalus*, 117:213–239, 1988.

[26] Anya Hurlbert and Tomaso Poggio. A network for image segmentation using color. In *Proceedings of the Denver Conference on Neural Networks*, Denver, CO, 1988.

[27] Anya Hurlbert and Tomaso Poggio. Synthetizing a color algorithm from examples. *Science*, 239:482–485, 1988.

[28] D.P. Huttenlocher and S. Ullman. Object recognition using alignment. In *Proceedings of the 1st International Conference on Computer Vision*, pages 102–111, London, England, June 1987. IEEE, Washington, DC.

[29] David W. Jacobs. The use of grouping in visual object recognition. A.I. Technical Report No. 1023. Artificial Intelligence Laboratory, Massachusetts Institute of Technology, Cambridge, MA, October 1988.

[30] Hsien-Che Lee. Method for computing the scene-illuminant chromaticity from specular highlights. *Journal of the Optical Society of America*, 3:1694–1699, 1986.

[31] Hsien-Che Lee. Estimating the illuminant color from the shading of a smoth surface. A.I. Memo No. 1068. Artificial Intelligence Laboratory, Massachusetts Institute of Technology, Cambridge, MA, August 1988.

[32] Donna M. Fritzsche Lewis W. Tucker, Carl R. Feynman. Object recognition using the Connection Machine. In *Proceedings IEEE Conf. on Computer Vision and Pattern Recognition*, Ann Arbor, MI, June 1988.

[33] James J. Little, Heinrich H. Bülthoff, and Tomaso Poggio. Parallel optical flow using local voting. In *Proceedings of the International Conference on Computer Vision* Tarpon Springs, Florida, December 1988. IEEE, Washington, DC.

[34] David G. Lowe. *Perceptual Organization and Visual Recognition*. Kluwer Academic Publishers, Boston, 1986.

[35] Jose L. Marroquin *Probabilistic Solution of Inverse Problems*. PhD thesis, Massachusetts Institute of Technology, Cambridge, MA, 1985.

[36] Jose L. Marroquin, Sanjoy Mitter, and Tomaso Poggio. Probabilistic solution of ill-posed problems in computational vision. In L. Baumann, editor, *Proceedings Image Understanding Workshop*, pages 293–309, McLean, VA, August 1985. Scientific Applications International Corporation.

[37] C. A. Micchelli. Interpolation of scattered data: Distance matrices and conditionally positive definite functions. *Constr. Approx.*, 2:11–22, 1986.

[38] Barbara Moore and Tomaso Poggio. Representation of properties of multilayer networks. In *Proceedings of the International Neural Network Society Annucl Meeting*, Boston, MA., September 1988.

[39] T. Poggio, W. Yang, and V. Torre. Optical flow: Computational properties and networks, biological and analog. In *The Neuron as a Computational Unit Proceedings*, Cambridge, UK, June 1988. Scientific Applications International Corporation.

[40] Tomaso Poggio. Computer vision. In E. Clementi and S. Chin, editors, *Biological and Artificial Intelligence Systems*, pages 471–483, Leiden, 1988. ESCOM Science Publishers.

[41] Tomaso Poggio. Learning, regularization, and splines (abstract). In *Proceedings of the International Neural Network Society Annual Meeting*, Boston, MA,, September 1988.

[42] Tomaso Poggio and C. Koch. Ill-posed problems in early vision: from computational theory to analog networks. *Proceedings of the Royal Society of London B*, 226:303–323, 1985.

[43] Tomaso Poggio, J. Little, E. Gamble, W. Gillett, D. Geiger, D. Weinshall, M. Villalba, N. Larson, T. Cass, H. Bülthoff, M. Drumheller, P. Oppenheimer, W. Yang, and A. Hurlbert. The MIT Vision Machine. In *Proceedings Image Understending Workshop*, Cambridge, MA, April 1988. Morgan Kaufmann, San Mateo, CA.

[44] Tomaso Poggio and Alessandro Verri. Motion field and optical flow: Qualitative properties. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 1988 (in press).

[45] Tomaso Poggio, Harry Voorhees, and Alan Yuille. Regularized solution to edge detection. *Journal of Complexity*, 4:106–123, 1988.

[46] M. J. D. Powell. Radial basis functions for multivariable interpolation: a review. In J. C. Mason and M. G. Cox, editors, *Algorithms for Approximation*, Clarendon Press, Oxford, 1987.

[47] K. Rinner. Studien ueber eine allgemeine, vorraussetzungslose loesung des folgebildanschlusses. *Oesterreichische Zeitschrift fuer Vermessung*, 23, 1963.

[48] T. D. Sanger. Optimal unsupervised learning. *Neural Networks*, 1(S1):127, 1988. Proc. 1st Ann. INNS meeting, Boston, MA

[49] Karen B. Sarachik. Characterising an indoor environment with a mobile robot and uncalibrated stereo. In *IEEE Conference on Robotics and Automation*, Scotsdale, AZ, May 1989.

[50] Karen B. Sarachik. Simple map building for a mobile robot with uncalibrated stereo. In *AAAI Spring Symposium on Navigation*, Stanford, CA, March 1989.

[51] Eric Saund. Symbolic construction of a 2d scale-space image. A.I. Memo No. 1028, Artificial Intelligence Laboratory, Massachusetts Institute of Technology, April 1988.

[52] B.V.H. Saxberg. Projected free-fall trajectories i. theory and simulation. *Biological Cybernetics*, 56:159-175, 1987.

[53] Steven A. Shafer. Using color to separate reflection components. *Color Research and Applications*, 10(4):210-218, 1985.

[54] J.E. & C.A. Mead Tanner. In *VLSI Signal Processing II,(Proceedings of the ASSP Conference on VLSI Signal Processing)*, pages 36-47. 1986.

[55] A. N. Tikhonov and V. Y. Arsenin. *Solutions of Ill-posed Problems*. W.H.Winston, Washington, D.C., 1977.

[56] Shimon Ullman. Visual routines. *Cognition*, 18, 1984.

[57] Shimon Ullman. An approach to object recognition: Aligning pictorial descriptions. A.I. Memo No. 931, Artificial Intelligence Laboratory, Massachusetts Institute of Technology, December 1986.

[58] Shimon Ullman and Amnon Sha'ashua. Structural saliency: The detection of globally salient structures using a locally connected network. A.I. Memo No. 1061, Artificial Intelligence Laboratory, Massachusetts Institute of Technology, July 1988.

[59] Harry Voorhees and Tomaso Poggio. Computing texture boundaries from images. *Nature*, 333(6171):364-367, 1988.

[60] Daphna Weinshall. Seeing "ghost" solutions in stereo vision. A.i. memo no. 1073, Artificial Intelligence Laboratory, Massachusetts Institute of Technology, September 1988.

[61] A. Yuille and N. Grzywacz. The motion coherence theory. In *Proceedings of the International Conference on Computer Vision*, pages 344-354, Washington, D.C., December 1988. IEEE Computer Society Press.

USC IMAGE UNDERSTANDING RESEARCH: 1988-89 [*]

R. Nevatia, K. Price, and G. Medioni
Institute for Robotics and Intelligent Systems
University of Southern California
Los Angeles, California 90089-0273

ABSTRACT

This paper summarizes the USC Image Understanding research projects and provides references to more detailed sources of information. Our work has focused on the topics of: robotics vision, mapping from aerial images, motion analysis, some general techniques and parallel processing.

1 INTRODUCTION

This paper summarizes our research projects during the past year. Some of this work is described in more detail in other papers in these proceedings[1, 2, 3, 4]; this work is covered only briefly in this summary. We also provide references to details for work not described elsewhere in these proceedings.

Our research activity has focussed on the following major topics:

- Robotics Vision
- Mapping from Aerial Images
- Motion Analysis, and
- Parallel Processing

2 ROBOTICS VISION

Our concentration here has been on description of 3-D shape and recognition of objects based on shape. We have made some major progress in these areas in the last year. We have studied techniques using range data, a pair of stereo images, and a single intensity image. We have also constructed some range finders in our laboratory. We have been developing methods for both surface and volume descriptions. Both methods rely on the same underlying philosophical concepts - that complex shapes need to be described by decomposition into "simpler" parts and that the inter-relations of the parts are a significant aspect of the shape description. The decomposition can be carried out successively to the desired level of detail. We call such descriptions *structured, hierarchical descriptions.*

2.1 RANGE DATA UNDERSTANDING

Range imagery provides an interesting domain of application in which the measurables in the image relate directly to the *shape* of the objects in the scene (as opposed to their texture, color, reflectance, etc. in intensity imagery). Our goal is to perform high level tasks such as object recognition and pose identification. Such geometric reasoning can only be performed using object-centered descriptions. We briefly outline the advances we have made in designing range finders to acquire depth images, in generating object-centered descriptions, and in recognizing complex objects in scenes with substantial occlusion.

[*] This research was supported, in part, by the Defense Advanced Research Projects Agency contracts DACA76-85-C-0009 and F33615-87-C-1436, order No. 3119 and monitored by the U.S. Army Engineer Topographic Laboratories and Air Force Wright Aeronautical Laboratories respectively.

Range finders

We have two different range finding systems available to generate a range map of a given 3-D object, both of them based on active triangulation. The first consists of an independent laser system generating a sheet of light projected on the target object, which is placed upon a translation or a rotary table driven by a personal computer. This computer includes a video digitizer board to which two CCD cameras, looking at the scene from both sides of the sheet of light, are connected. For each camera, the projection of the 2-D curve consisting of the intersection of the laser plane with the object is extracted. As our calibration procedure establishes the geometric transformation that back-projects any point of the image plane to the corresponding point in the laser plane, we are able to reconstruct the 2-D curve in the laser plane. Hence moving the object step by step provides a dense map of the surface of the object either in cylindrical coordinates (rotary table) or cartesian coordinates (translation table)[5].

Besides its low cost, this system has several advantages over similar existing systems. First of all, we use two cameras to limit the well known occlusion problem and we integrate range data obtained from these cameras into a single range image. The calibration of each camera is very simple, has a sub-pixel accuracy, and is performed only once as the laser and the camera do not move. Our data acquisition uses an interpolation technique that produces very accurate depth measurements (typically 0.2mm precision at 1m) and our system also provides intensity data in registration with the range data.

In the case where we can not or do not wish to move the scene on a tray, we use a system that consists of a nematic liquid crystal mask inserted into a slide projector to provide an illumination pattern and a CCD camera looking at the scene from a different angle. The hardware was provided courtesy of Prof. S. Inokuchi from Osaka University, and the details of the system can be found in [6]. The main advantage of this system is speed, since by projecting a set of n Gray-coded patterns onto the scene, we obtain depths for 2^n lines.

Generating surface descriptions

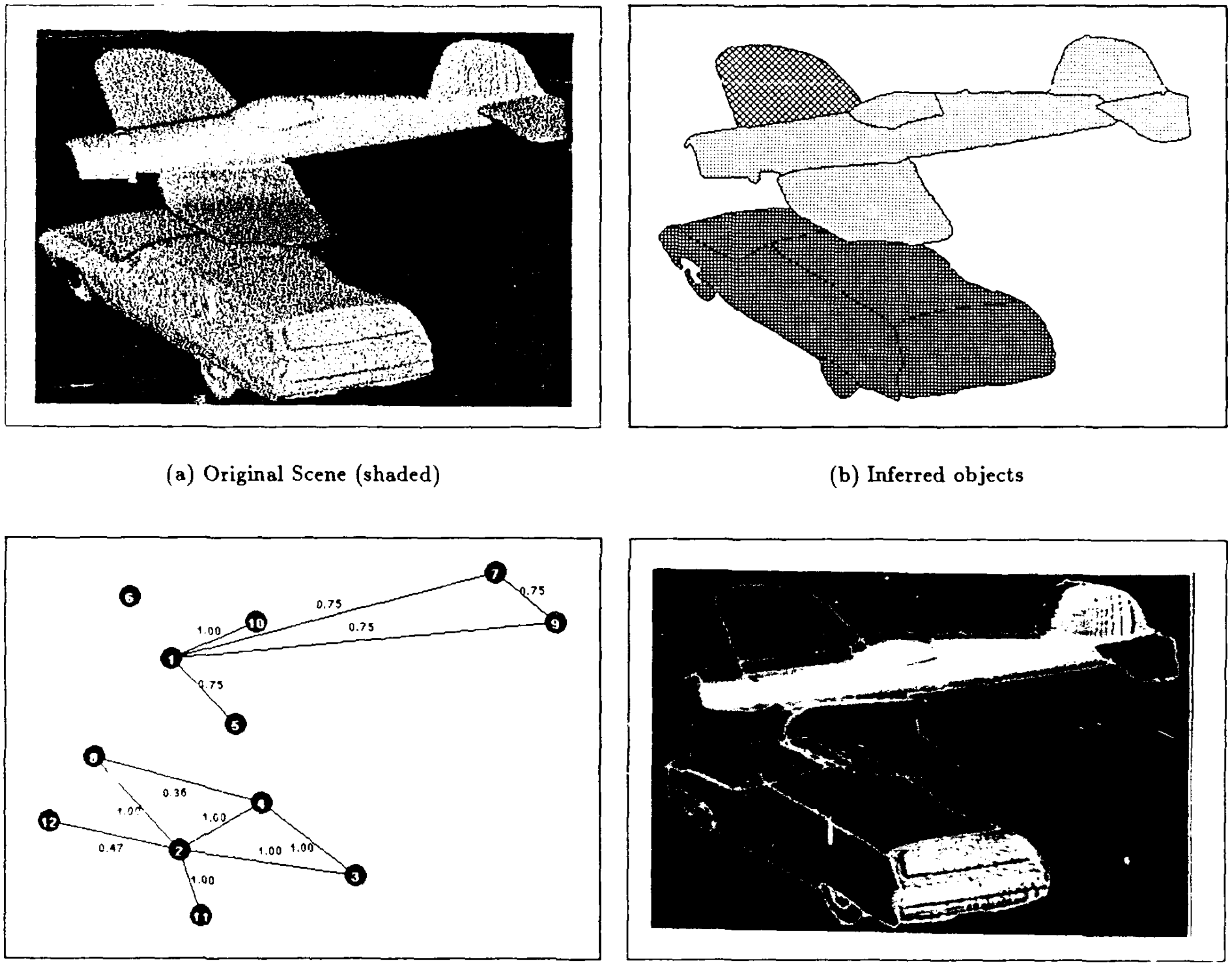
In order to obtain useful surface descriptions, we need not only to devise a proper formalism using the criteria of *richness stability* and *local support*, but also to design proper implementation tools to deal with real images (noise, quantization and digitization).

We have chosen to segment range images into simple surface patches, whose boundaries correspond to surface discontinuities (C_0) or surface orientation discontinuities (C_1). Each surface patch is then globally approximated by a bivariate quadratic polynomial. The details can be found in [7]. This segmented representation of a scene may be viewed as a graph whose nodes capture information about the individual surface patches and whose links represent the relationships between them, such as occlusion and connectivity. Simple reasoning on these relationships is used to decompose the full graph into disjoint subgraphs corresponding to different objects. An example is shown in figure 1a-c.

The success of this representation critically depends on our ability to compute the necessary attributes, such as gradients and curvature, from an image in the presence of noise. We have found adaptive smoothing to be a tool of great value for such operations. The details can be found in [8, 9], but the ideas can be summarized as follows: *The general purpose of our Adaptive Smoothing scheme is to smooth a signal - whether it is an intensity image, a range image or a planar curve - while preserving and even enhancing its discontinuities.* This is achieved by repeatedly convolving the signal with a very small averaging filter modulated by a measure of the signal discontinuity at each point. A relatively small number of iterations is needed to obtain a smooth signal suitable for features extraction. In range images, we use curvature features such as curvature extrema or zero-crossings which are easily detected and directly localized after Adaptive Smoothing as opposed to Gaussian Scale-Space approaches where a tedious tracking procedure is needed.

3-D Object Recognition

We have been able to use the above descriptions to achieve successful recognition of complex objects in scenes containing multiple objects that are only partially visible and are occluding each other. An example of recognition is presented in figure 1(d), and a detailed treatment can be found in [10, 11]. For the purpose of matching, a model is represented by a set of such descriptions from multiple viewing angles, typically 4 to 6. Models can therefore be acquired and represented automatically. Matching between the objects in a scene and the models is performed by three modules: the *screener*, which finds the most likely candidate views for each object, the *graph matcher*, which performs a detailed comparison between the potential matching graphs and computes the 3-D transformation between them, and the *analyzer*, which takes a critical look at the results and proposes points to split and merge object graphs.



(a) Original Scene (shaded)

(b) Inferred objects

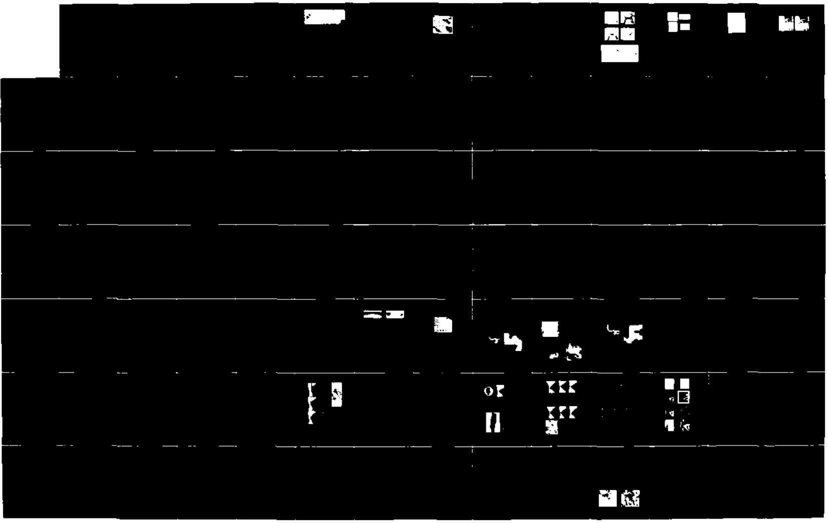
(c) Graphs

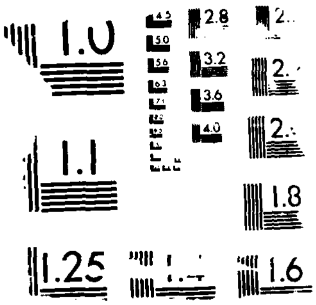
(d) Results

Figure 1: Segmentation of a complex range image.

AD-A212 981   PROCEEDINGS OF IMAGE UNDERSTANDING WORKSHOP HELD IN PALO ALTO CALIFORNIA O..(U) SCIENCE APPLICATIONS INTERNATIONAL CORP MCLEAN VA   L S BAUMAN MAY 89   2/13

UNCLASSIFIED   N00014-86-C-0700   F/G 17/11   NL




1.0
4.5
5.0
5.6
6.3
2.8
3.2
3.6
4.0
2.2
2.5
2.0
1.1
1.8
1.25
1.4
1.6

Current focus

Our interest is in using our current range understanding sytem as a core and in extending it in the following ways:

- Other representations
  The current system can only handle cartesian maps, whereas different sensors may furnish data more easily expressed in cylindrical or spherical coordinates.
- Automatic model acquisition
  The current model representation by a set of partial views should be modified to describe the entire object by a single graph. This procedure should be done automatically by merging the different views of an object. We are investigating this merging both at the data and symbolic level.
- Parameter sensitivity
  The segmentation of range images into a set of surface patches depends on a few thresholds, similar to the ones used in edge detection. This dependence can be significantly reduced by observing the distribution of errors between the original data and the best surface fit. If the errors are not randomly distributed, the patch can be resegmented with tighter parameters.
- More general patches
  The current method can handle a large variety of complex objects, under the condition that the boundaries between the patches are sharp. When the object resembles a free form surface, the approach used here breaks down, because it forces us either to accept a bad polynomial approximation, or to introduce phantom boundaries to preserve the goodness of fit.

2.2 SHAPE DESCRIPTIONS FROM INTENSITY IMAGES

Deriving reliable shape descriptions from one or more intensity images is much more difficult than from range data because the information we can extract is sparse and imperfect. Our feature extraction methods do not find all of the object boundaries and many of the boundaries may correspond to shadows, surface markings and noise. Thus, our description methods must distinguish between various kinds of boundaries in addition to generating the shape descriptions themselves, *i.e. the segmentation* and the shape description problem can no longer be separated.

We have developed two separate systems for this task. Both use a representation for object shape that assumes that the objects of interest have some symmetries and that segmentation and description can be achieved by finding these symmetries, which can be thought of as resulting from the projections of generalized cones. Generalized cones have been used extensively in previous shape analysis work, usually, however, perfect data, such as that available from a range finder, is assumed. In earlier work, we described a technique to work with sparse and imperfect data that assumed that the scene contained a class of objects known as *linear, straight, homogeneous generalized cylinders* [12]. Our new method is able to handle much more general objects.

Our technique represents a complex object by decomposition into simpler parts. Our task now is to find which boundaries outline a part and how the parts are inter-related. We do this based on a simple observation:

object parts are of finite extent, hence they must be terminated by a boundary or by another part.

Our initial hypotheses as to where object parts may be found is based on finding symmetrical pairs of boundaries which could constitute the "axial contours" of a generalized cylinder (or a "ribbon"). Even in simple scenes, this can produce hundreds of possible alternatives. The choice between these multiple hypotheses is made based on the above observation, namely that real parts must terminate somewhere. This process is highly effective in narrowing the list of hypotheses and also produces hierarchical, segmented descriptions for complex objects simultaneously. This method is described in more detail elsewhere in these proceedings[3].

In another approach, we attempt to implement a process of "perceptual grouping." Again, we use the symmetry of contours to form initial hypotheses. The choice between multiple hypotheses is based on a process of competition between them. Competition is implemented in a highly parallel "constraint satisfaction network." Further geometric reasoning is applied between the hypotheses that survive this competition. Good segmentation has been obtained for a variety of highly complex scenes in our experiments. Yet higher levels of reasoning can be applied if a pair of stereo images are available resulting on complete 3-d description of visible parts of the objects in the scene. This technique is described in detail later in these proceedings[4].

2.3 DEPTH FROM STEREO

Our previous stereo methods [13, 14] relied on segments derived from connected edgels as primitives. In the presence of texture, however, this continuity along segments cannot be enforced, since the segments tend to be very short and

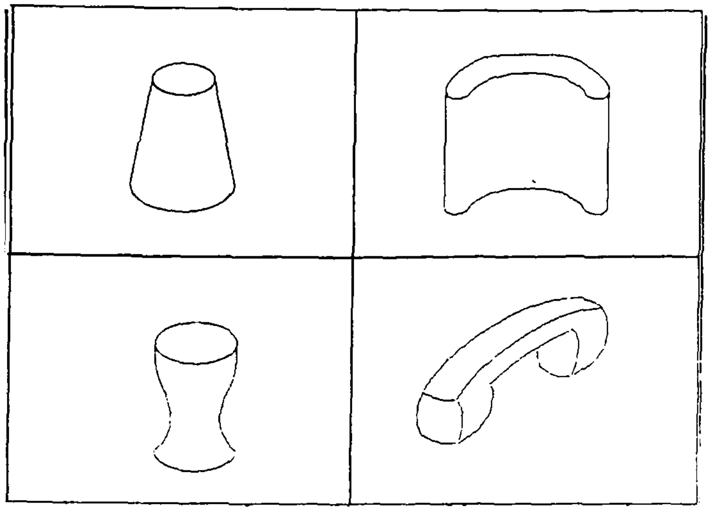


Figure 2: Some figures for which we readily perceive 3-D shape from contour alone.

disconnected. We have made significant progress in building a system that aims at robustness with respect to scene characteristics, from textured outdoors scenes to highly regular man-made objects. It offers the advantages of both area-based (dense map) and feature-based processing (accurate disparity) by combining them wherever possible. In the current version, the area-based process occurs first and is refined by the integration of edge information. It is based on our observation that whenever there is enough "texture" (measured as intensity variation in a small window), then a correct correspondence can be obtained by a local process. The area based approach proceeds by computing a texture for each image view and performing a simple cross correlation between them. A match is accepted if it corresponds to a peak for both views and this peak is high enough. The resulting dense disparity image containing a few holes and incorrect matches is then filtered using the smoothness assumption to fill small gaps and remove small spikes. Note that contrary to the case of feature based stereo, this smoothness assumption is justified since we reason about patches of opaque objects, and that we can make inferences about occlusion and detect "penumbral" areas (visible in only one of the views). This disparity map is a smoothed version of the true one, however, because of the finite width of the windows used in processing. The problem is most acute at C_0 (depth) and C_1 (crease) discontinuities, but can be solved by introducing the edge information: the disparity map is adaptively smoothed [9] subject to the constraint that the disparity at edgels is fixed. It is important to note that this method gives an active role to edgels parallel to the epipolar lines, whereas they are discarded in most feature-based systems. We have obtained very good results on complex scenes in different domains, as shown in the paper included in these proceedings [1].

2.4 SHAPE FROM CONTOUR

In another project, we are investigating how the 3-d shape of an object can be inferred from its contour in a single view. Most previous work in this area has been in the domain of polyhedral objects with only some limited techniques available for analysis of curved surfaces [15, 16, 17]. We have been studying objects such as shown in figure 2. Our basic assumption is that much of an object's 3-d shape is given by the symmetries in the figure and that non-symmetric figures give poor 3-d impressions to humans as well.

Our technique uses certain constraints for determining the orientations of the points on the surface in 3-D based on the following assumptions:

- Observed skew symmetries in the image correspond to real symmetries in 3-D
- That a certain contour in the image is planar

In [18] we have shown that in some cases these constraints are sufficient to uniquely determine the 3-d surfaces from contour alone. However, we view these techniques as still being preliminary with much more analysis needed for complex shapes. In particular, we need to examine if the assumptions hold in all cases of complex shapes.

The techniques for determining shape from contour, including our method outlined above, usually assume that the image is obtained by orthographic projection. We have also been working on generalizing the techniques to work with perspective projection. We find that some of the equations are more complex but the orthographic techniques still apply. One surprising result of our analysis is that in some cases the perspective analysis actually gives tighter constraints than orthographic analysis. These techniques are described in [19].

3 MAPPING FROM AERIAL IMAGES

Our goal here is to produce high-quality symbolic maps of complex, cultural scenes from aerial image data. For some time, we have been working with the domain of large commercial airport complexes. Such scenes have a variety of features such as the transportation network (runways, taxiways and roads), buildings (terminals, hangars, etc.) and mobile objects (airplanes, trucks, cars, etc.). Our aim is to produce descriptions of the individual objects in the scene as well as an integrated description of the entire scene including the functional relationships between the parts.

Our motivation for this work is two-fold. Firstly, the specific tasks are of great practical significance for a variety of applications. Secondly, we believe that the problem domain provides a rich testbed for experiments in building high-performance visual "expert" systems. We do not necessarily imply that the exact algorithms developed for this task will also be useful for all other tasks, but merely the hope that the approach will carry over for similar tasks. We also believe that experience with specific domains is essential to development of more generic vision systems. Mapping requires dealing with a multiplicity and variety of objects in a natural environment that contains texture and markings. The solution requires use of powerful "bottom-up" descriptive techniques as well as the use of domain knowledge. Such capabilities are obviously going to be needed by vision systems in other domains also.

Our approach in the design of the system is that it must be modular and that the modules interact mostly at high, symbolic levels. In airports, for example the modules may be for detecting and describing the transportation network, the buildings and the mobile objects. Detection of one type of object, such as a taxiway, may aid in increasing the confidence of a structure believed to be a passenger terminal (and *vice-versa*). However, we believe that such interaction takes place at a high level, after symbolic, object level hypotheses have been formed. This process can be considered hierarchical; each module has sub-modules that operate in a similar way. Thus, the transportation network module may consist of runway, taxiway and road modules; each of which operates somewhat independently but uses context provided by the detection of other structures. Some structures may be more prominent and easier to detect, for example, runways are easier to detect than taxiways. In that case, the former provides the context for detection of the latter.

In our analysis, we do not assume specific knowledge of the scene, such as would be given by a detailed, current map of the specific airport complex. Instead, we only have *generic* information that the scene being viewed is an airport complex. Our approach is basically one of "hypothesize and verify." Various grouping operations relying on geometry, object shape and context form hypotheses that are then verified according to some desired attributes. Our system detects runways first, as they are more prominent and can provide the needed context for detection of taxiways (and many other objects in the scene), as these are much less constrained in shape and appearance than the runways. The converse is also true, *i.e.* finding taxiways connected to a runway can help increase the confidence of the detected runway.

In another project, we have been developing methods for detection and description of complex building structures [20, 21]. We have achieved what we believe is a major success in this effort and we are able to handle buildings with wings of different heights. The shapes are restricted to being compositions of rectangles, however. The key to the method is a technique for perceptual grouping of low-level features into meaningful high-level structures. This method is described in detail in a paper in these proceedings [4]. We expect that this technique can be generalized to work for a broad classes of objects, in aerial scenes and in other domains, and is a major focus of our current research.

Further validation of hypotheses should, ideally, take place in the context of the larger system that is also reasoning about other objects in the scene, such as the remainder of the transportation network, other buildings and the mobile objects. Location of these objects will mutually affect the confidence levels of the descriptions of other objects. Many interesting questions arise in the implementation of such interactions, such as the nature of the interaction and the order in which it takes place (*i.e* the control structure). We are investigating alternative techniques for this in our current work. The techniques described here should be viewed as a module for the larger system to operate with. Regardless of the fine structure of the larger system, it is our belief that the system needs modules which are fairly competent at finding the major, individual structures *without* the global context. The global context is useful to refine or confirm the initial hypotheses and in some cases to initiate new hypotheses but can not be a substitute for high quality description modules.

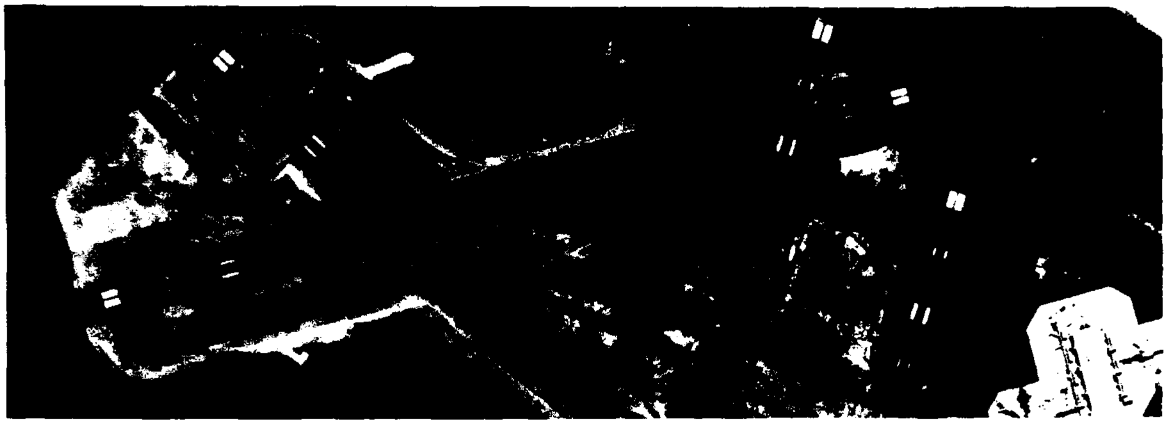


Figure 3: Portion of LOGAN scene

3.1 AN EXAMPLE

Previously we have reported on our work in the extraction of runways [22]. Our technique consists of hypothesizing runways by using linear segments, forming *anti-parallels* from them and then grouping the anti-parallels on the basis of continuity and collinearity. The verification of the hypotheses comes from detecting expected markings on the runways [23]. We have applied these techniques to a variety of airport scenes with success. In recent work, we have further enhanced our verification technique. In earlier work, all processing was performed at one resolution. This resolution is not adequate to detect all markings on the runway, on the other hand it is very expensive and unnecessary to process the entire image at the highest available resolution. For the task of verification of specific features, however, we can focus on just selected parts of the image and resegment it at the needed high resolution. Thus, we have a case of the results of higher level symbolic processing guiding the low-level segmentation on a second pass. We have found this technique to be highly effective in detecting subtle marks on the runway surfaces that increase our confidence in the detection of the runways.

Consider the scene shown in figure 3, a 2200 x 800 pixel image of a portion of Logan International Airport in Boston. The verified runway hypotheses are shown in figure 4. We are able to locate 71% of the centerlines, 63% of the sidestripes, 100% of the threshold and touchdown marks, 70% of the long distance marks, 56% of the short distance marks, and 89% of the blastpad marks. Additional markings are obtained by re-segmenting small portions of the image. We have tested this feedback step by looking for missing centerlines and blastpad marks. This brings the centerlines to 96% and the blastpad marks to 100% . Note from the markings detected in our example that the runways can be readily classified as precision instrument runways [24]. However, at this stage we do not attempt to specifically assign a confidence value to each detected runway.

Taxiways are much more complex objects than runways, as they can have a wider range in their geometrical parameters; they can be short or long, have a variety of widths, be straight or curved, and connect a variety of airport components. Taxiway detection is aided by the descriptions of previously detected runways, and also by knowledge of airport design [24]. We know for instance, the minimum acceptable distance between a taxiway and a runway if they are parallel, or the minimum angle that a taxiway may form with a runway. We also know that taxiways do not cross but join runways. Taxiway crossings however, are allowed.

The first step in detecting taxiways is to find long fragments which may correspond to fragments of taxiways. We select apars representing potential taxiway fragments in a manner analogous to the selection of potential runway fragments [22]: they have a range of widths, and either are parallel to a runway or, form an angle greater than 25° with a runway. Selected fragments are joined on the basis of continuity and collinearity, to form "long" straight hypotheses.

The second step attempts to extend these straight portions of taxiways. This work is in progress at this time; it includes the following context dependent processes:

1. Extension based on Aircraft Support: A large aircraft on a taxiway will cause the taxiway hypothesis to fragment, thus to extend the taxiway fragments, we first try to detect aircraft by looking for symmetries due to the aircraft wings and fuselage at each fragment end. If an aircraft is detected, the taxiway is extended the length of the aircraft.

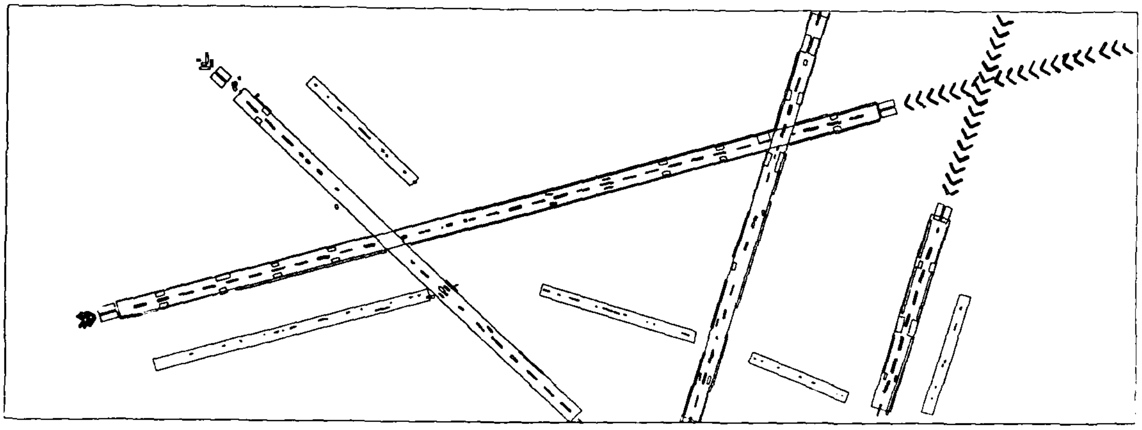


Figure 4: Verified runways and taxiways

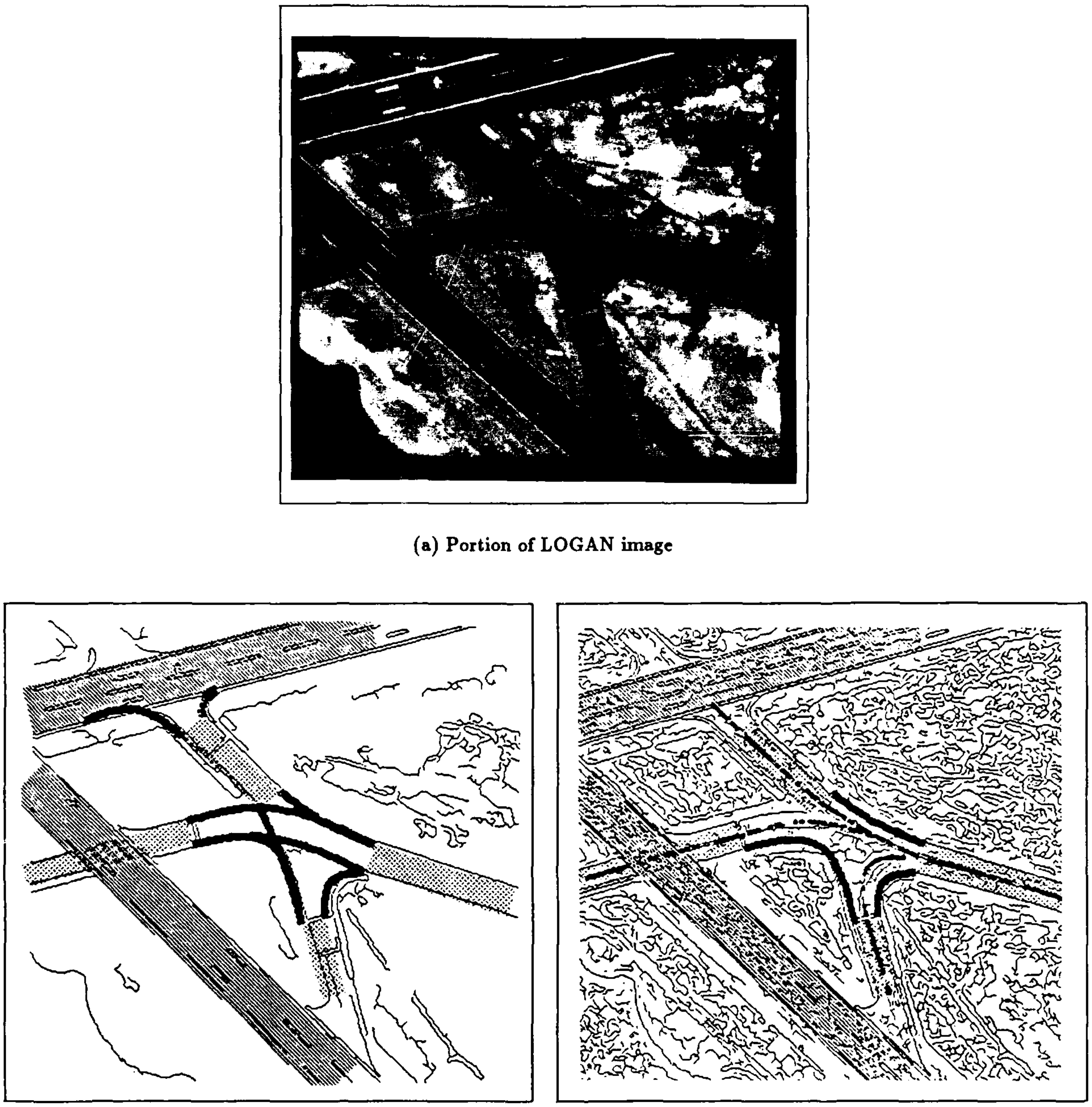
2. Extension based on Runway Context: We attempt to extend or discard taxiway hypotheses fragments based on their spatial relationships to verified runways in the scene. The following steps are taken:
   (a) Fragment intersects runway: The taxiway hypothesis fragments are extended until they intersect a runway. If the intersection angle is greater than the minimum intersection angle and the distance between the taxiway hypothesis fragment and the runway intersection point is small, we look for additional evidence to extend the fragment into the runway. This evidence includes checking for shorter apars collinear to the taxiway in this region and, failing this, the detection of aircraft in this region. If we find sufficient evidence, the taxiway hypothesis is extended into the runway.
   (b) Fragment is parallel to runway: If the taxiway fragments are parallel to one of the verified runways, we look for small wide apars joining the end of the taxiway fragment to the runway indicating the presence of a taxiway apron.
   (c) Extension based on Taxiway Intersection: (see below)
   (d) Extension based on Resegmentation: It is possible that a material change in the taxiway caused problems for the initial grouping processes. We attempt to extend the taxiway fragments by resegmenting image windows extending beyond the fragments' ends, and looking for evidence of taxiway continuation. This process is continued until no further evidence is found. At this point, we repeat steps (a) and (b).

The apars representing hypotheses of straight portions of taxiways are shown in figure 4. In this result, only process 1 was applied. Extension of taxiways based on intersections (process 2c) attempts to describe the junctions and connections among taxiways and between taxiways and runways. The accurate description of the junctions between pathways also helps determine their function. Some are used as holding aprons; some are exit ramps (the closer to the end of the runway, the smaller the angle between them, with angle determining the allowed exit speed); some are merely connecting pathways; the continuous centerline determines the "legal" turns and paths; and so on.

The image in figure 5a, a portion of the image previously shown in figure 3, shows the intersection of four taxiways, and connections between taxiways and runways when these are not parallel to each other.

We describe the junctions by explicitly locating the boundaries, or portions of the boundaries, of the sections of roadways that connect the previously detected runways and straight portions of taxiways. We use the geometrical relationships among these to compute the size and shape of the search windows where we look for the boundaries. Our method distinguishes two types of junctions: L-junctions (typically between portions of taxiways), and T-junctions (typically joining taxiways and runways). More complex junctions, such as the one in figure 5, are viewed as overlapping L-junctions. Note that there are no junctions between crossing runways; they are considered overlapping.

For each pair of potential "joinable" (nearby and converging) fragments we distinguish an "inside" and an "outside" boundary. The inside boundary, if it exits, would be found on the side where we measure the smaller angle between the two elements. On the other hand, T-junctions are considered to have two "inside" boundaries. A second classification involves the boundaries themselves. Some are curved while others are straight. The curved boundaries — in airport design — actually consist of circular or parabolic sections. However, we model the straight boundaries as two straight lines, and the curved boundaries by means of cubic splines. For each boundary we apply both models and then the choose the better fit.



(a) Portion of LOGAN image

(b) Selected connections

(c) Underlying edges and evidence of markings

Figure 5: Taxiway junction at LOGAN

We look first for the inside boundary, and then for the outside boundary. (There is always an inside boundary. However, at complex intersections there may be no outside boundaries.) If there is no evidence of an inside boundary, we do not look for an outside boundary. The method is as follows:

1. For each pair of joinable elements collect the context information to compute a search window for the inside boundary.
2. Look for an inside boundary. We compute a series of splines using three points for each: two anchor (fixed) points (at the ends of taxiway fragments or inside runways) and a sliding point (which moves approximately along the bisector of the two elements). For each spline, we compute the overlap of the spline with the underlying intensity edges. The spline that returns the highest number of edges is taken as a hypotheses (possible inside boundary) if the following criteria are met:
   (a) The length of the underlying boundary (or boundary fragments) is at least one half of the length of the spline. In other words, allow 50% boundary fragmentation and/or partial spline-to-boundary fit.
   (b) The "junction" between the spline and the element boundaries is smooth (15° tolerance), *i.e.*, the tangent to the spline at the intersecting edge points is similar to the direction of the edge.
3. Compute a search window for an outside boundary using information from the detected inside boundary.
4. Look for outside boundary. A process similar to that for inside boundaries. Compute a series of splines using two anchor points and a sliding guide point. We compute the intersection of each spline with the underlying intensity edges. The spline that returns the highest number of edges is taken as a hypotheses (possible outside boundary) if similar criteria are met.

Figure 5b shows the original underlying edges, the taxiway and runway context (shaded areas), and the selected connections we compute. As before, verification consists of finding the markings we expect. Our method looks for the centerlines along he roadway as follows:

1. Re-segment and collect context information: We resegment the image to include all intensity edges in the neighborhood of the junction.
2. Compute the search window and look for centerline edges. The search process is similar to that for inside boundaries. We compute a series of splines using two anchor points and varying the position of the guide point. We also look for the certerline edges along the straight portions of taxiways.

Figure 5c shows the re-segmented edges, the inside boundaries used to help locate the evidence of centerlines, and the evidence found. Further details and examples are given in [25].

We have developed two techniques to detect buildings. One [20] is based on shape constraints for detection, and the shadows that buildings cast for verification and obtaining three dimensional information. The second [21, 4], is more suitable for complex structures. It uses perceptual organization to generate multiple building hypotheses from a stereo pair of images. Next, promising hypotheses are selected by a constraint satisfaction network based on Hopfield's model. The selected hypotheses (rectangular subparts) are then processed by stereo to compute three dimensional descriptions. An example is given in figure 6. Figures 6a,b show a stereo pair of a terminal building at Logan Airport. The rectangles selected and matched by stereo are shown in figures 6c and d respectively. Figure 6e shows a rendered view of the building generated from an arbitrary view point using the 3D information computed.

We have been working with airport scenes representing extremes of complexity that are encountered in *major* commercial airports (smaller airports are much easier to analyze). These represent a wide spectrum of runway types and conditions; different runway surface materials, homogeneous and non homogeneous surfaces; runways with shoulders of the same or different material and of various widths; and so on. The performance of the technique shows a high degree of reliability if good image quality and adequate resolution are available.

We believe that the results that we have obtained indicate very good performance and indicate that the method will work well on other examples. Also, it must be realized that it is not our contention that the various objects can be analyzed in isolation. Their detection and description is dependent on the various objects in the scene. Interaction among such objects is part of our current research. We do believe that the results that we can obtain indicate that our methods will provide very high quality input to the larger system.

4 MOTION ANALYSIS OVERVIEW

We have a number of ongoing efforts in the analysis of sequences of images including analysis based on a moving observer and the detection and analysis of moving objects. Autonomous navigation provides the context for much of the work, though the techniques are of much broader utility. The effort is supported by our "Knowledge-based Vision Techniques" contract as part of the DARPA strategic computing program.

Motion analysis using long range or feature point analysis techniques forms the central focus of our work. This approach involves extracting a set of consistent features from a sequence of images (lines, corners, contours, regions, etc.), finding the corresponding features in consecutive frames of the sequence by a series of frame to frame matching operations, and finally computing three-dimensional motion estimates based on the series of correspondences, which also produces depth information for the feature points. These problems have been addressed separately and to a lesser extent together in a combined system. Our work on spatio-temporal analysis and merging a series of depth maps does not fit directly in this scheme, but both do use a feature based analysis to guide the processing and improve the results. This overview discusses the current status of the research in these areas.

4.1 FEATURE MATCHING IN MOTION IMAGES

Elsewhere in these proceedings [2] we describe a method to establish correspondences between images in a motion sequence. The matching method is similar to our previously reported work[26], but we now use multiple scales and better primitives. We first smooth the images with adaptive filters at different scales, then detect edgels, link them and extract segments and super-segments at each scale. We then match the detected features hierarchically, starting with the larger scale, using an extension of our matching algorithm, in the following way (assuming the higher mask is h and the lower is l):

1. Match the two images using features detected by mask h.
2. Match the results of the same image smoothed by h and by l for both images.
3. Combine the results to obtain *predicted matches* for the images smoothed by l.
4. Match the images smoothed by l using the predictions.

Advantages

1. **Edge localization:** Hierarchical Smoothing with adaptive filters at different scales yields a hierarchical set of features. But, unlike zero crossings of Laplacian of Gaussian masks (the features previously used), they shift the edges by no more than a pixel between scales, thus matching the same image for different scales becomes trivial. Any edge detector can be applied to the smoothed images, since they are nearly piecewise constant and free of noise. We have used Canny's edge detector to obtain edges that now represent real events and are accurately located (unlike zero crossings).
2. **Hierarchical Matching** Matching images smoothed by large masks is relatively easy, as only a few strong features are preserved. These are later used to restrict and guide matches for finer masks, thus reducing both errors and computation time.

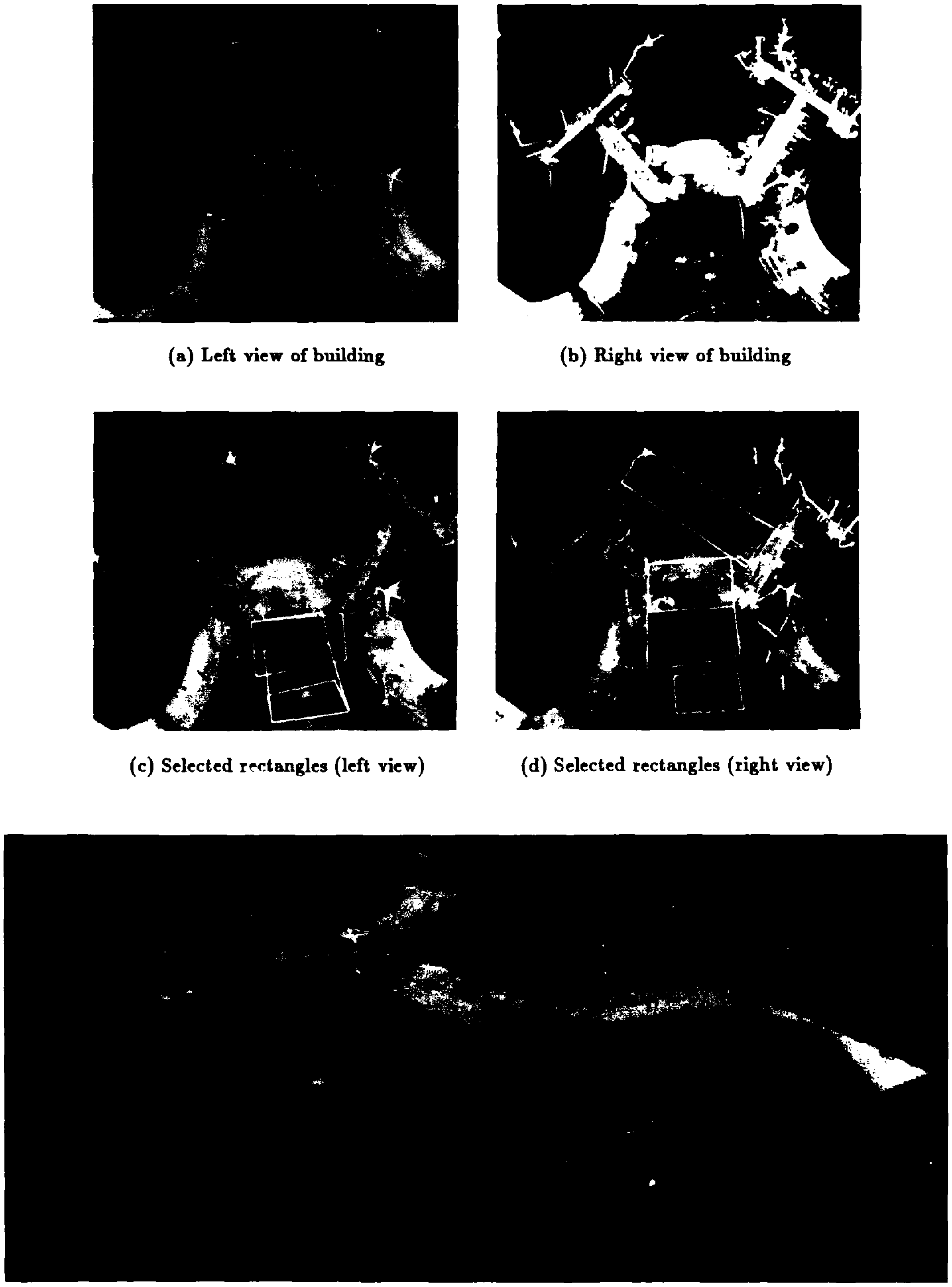
Applying this method to real images produces very good results. We have been able to match less closely spaced frames yet obtain better results than with previous approaches.

4.2 CHRONOGENEOUS MOTION

Our earlier work in motion estimation from matching points assumed that the motion was constant through the several (e.g. 5) frames[27]. By developing a technique that includes time along with 3-D position we have derived a Chronogeneous Motion estimation technique[28]. This allows for the description of the constant motion cases assumed by most researchers, as well as some cases of accelerated motion. This technique builds on our earlier work in multi-frame motion estimation and the derivations both extend and confirm the earlier theoretical development. We have implemented a basic 3D structure from motion with acceleratior case. This case has a closed form solution given point correspondences in multiple frames. Such a soultion is not possible for the general case of chronogeneous motion. The case of structure from known motion or motion from known structure is much easier and also has a closed form solution. This method is being tested on synthetic data to determine the effects of quantization error and noise in the input data.

4.3 NAVIGATION

One task for an autonomous land vehicle (ALV) is to use sensory data such as range and/or color images for visual navigation. The vehicle has an inertial system that provides a good estimate of the vehicle position and orientation with respect to a world coordinate system. As these measurements may deviate while the vehicle is moving, one task of the vision system is to correct the estimate of the vehicle position.



(a) Left view of building

(b) Right view of building

(c) Selected rectangles (left view)

(d) Selected rectangles (right view)

(e) Rendered 3-D view of building

Figure 6: Terminal building at LOGAN

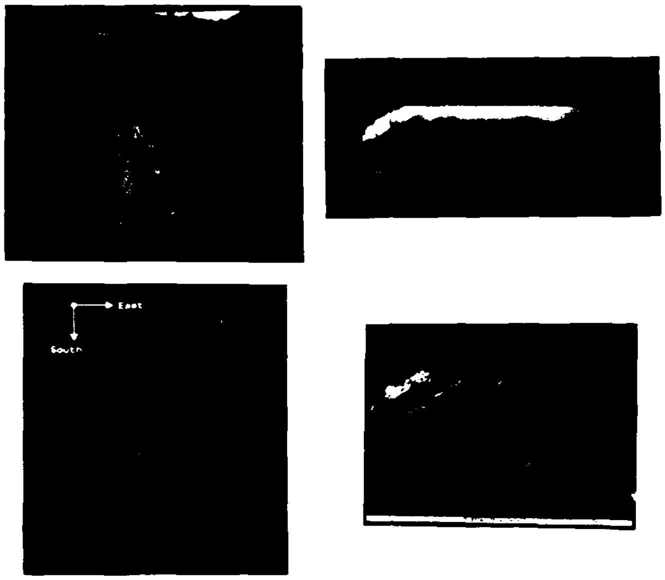


Figure 7: Local Cartesian Elevation and Color Maps.

We chose to work in the context of an unknown environment using range *and* color images obtained from Martin Marietta Aerospace. Knowing the position and orientation of the sensors with respect to the vehicle coordinate system along with their parameters, we first established the geometric transformation that links the range data to the color data. Thus, given a point in the color image where the fields of view of the two sensors overlap, it is possible to find the 3-D coordinates of that point using a ray-tracing technique and the range image. Hence, the output of an edge detection operator applied to the color image can be transformed into a 3-D edge map. Updating the position of the vehicle thus consists first in matching the current 3-D edge map with the previous one, using the knowledge of the position of the vehicle from its internal sensors (this method is currently under development and gives promising results) to guide the matching process. The position of the vehicle is then updated by computing the 3-D transformation between matched 3-D edges using a least-square technique.

Whether or not we rely on the vision system to update the vehicle position, it is usefull for autonomous navigation purpose to generate a global cartesian elevation map (GCEM) of the area "explored" so far by the vehicle. For each position of the vehicle, we construct a local cartesian elevation map (LCEM) derived from the range data using a one pass technique. Along with the LCEM, we also derive a local cartesian color map (LCCM) by "painting" the LCEM using the same geometric transformations discussed above. Figure 7 provides results obtain for one position of the vehicle: the upper left frame shows the color image while the upper right frame shows the range image (after adjusting for the ambiguity interval). The lower left frame shows a down-looking view of the computed LCCM while the lower right frame shows a perspective view of the computed LCEM. The local maps can then be merged into a global map given the successive positions of the vehicle and the matching procedure used for adjusting the vehicle positions. Figure 8 shows a down-looking view of the computed global cartesian color map (GCCM) after merging successive local maps. The filled-in circles give the trajectory of the vehicle, with the world coordinate system shown in the upper left of the image.

4.4 MOTION FROM THE SPATIO-TEMPORAL VOLUME

Most approaches to motion analysis only use two or three image frames, therefore the estimates are unstable and noisy. In [29, 30], a method which shows how to utilize many frames is introduced. The principle behind this approach is to find the velocity components of an edge point along several different directions by taking *slices* in the temporal direction.

A *slice* is a collection of 1-D images of small width taken from successive frames in the sequence at the same position. This spatio-temporal data structure provides an easy way to trace a line segment through frames by finding *paths* in a slice. A *path* may be induced by any portion of an object boundary. When the object moves, the projection of

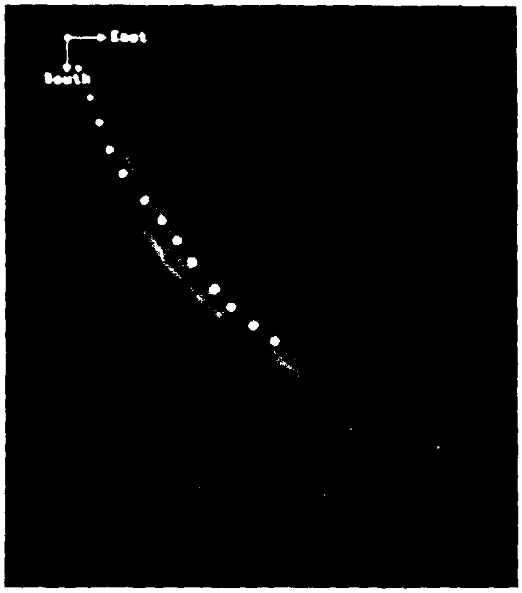


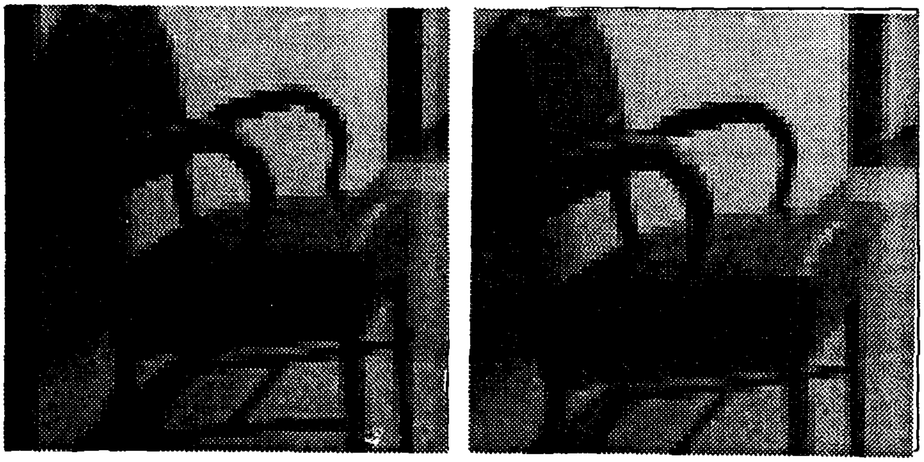
Figure 8: Global Cartesian Color Map.

the boundary sweeps out a surface in the 3-D spatio-t^emporal image volume, which is the image sequence with time as the third dimension. The slope of this path is used to compute the displacement in the direction parallel to the orientation of the slice.

The topology of paths in a slice gives important information to detect occlusion or disocclusion by extracting λ or Y junctions. The combination of results from multiple slices centered on a given point enables us to estimate its velocity in the direction normal to the tangent to the spatial curve. Each frame receives several messages from earlier image frames if there exists occlusion or disocclusion in the frame. Based on the location of occlusion or disocclusion and 8-connectivity, the segmentation of edge points into contours becomes easier because the ambiguity of the three way junction is resolved. The edge points in the first frame of the image sequence are then segmented into contours. From the normal flow field, the occlusion information gathered during *slice analysis*, and the spatial structure of the contours, the correct optical flow field induced by motion is then recoved under smoothness constraints.

Our programs work very well on synthetic image sequences but only achieve limited success on real image sequences, mostly because of fundamental problems in low-level processing. For this reason, we have now decided to choose smooth curves instead of edges as tokens. Smooth curves are generated by breaking linked contours where the tangent changes more than a certain threshold. Slices centered at the middle point of each smooth curve in both frames are taken and paths are extracted from those slices. Hash tables are set up with smooth curves as entries, associated with the correponding curves in the other frame linked by paths as the stored values. Unlike other methods, the correspondence among smooth curves with linking paths is rarely ambiguous. Smooth curves are grouped according to their connectivity in both frames. Now the global matching problem can be divided into smaller pieces, groups from both frames with paths linking their component curves are used to set up the global matching and velocity estimation. Figure 9 shows two frames from an image sequence with a chair inside the window. Figure 10 shows the smooth curves extracted from the two frames while matching groups are shown with the same texture.

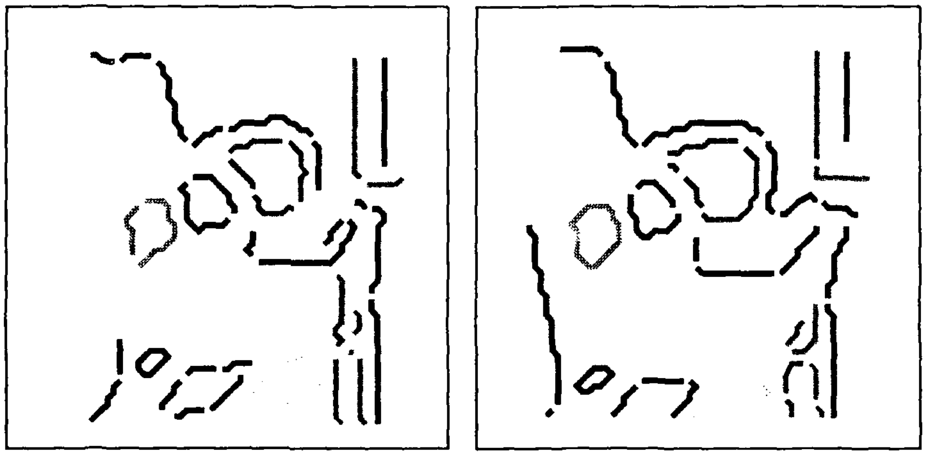
Next we intend to detect or confirm the existence of occlusion or disocclusion. Special attention is paid to curves joining at a junction or at the ends of a group of connected curves, these are places where occlusion or disocclusion (uncovering) may take place. Slices are taken at those places and analyzed to find if Y and λ junctions exist. We also intend to estimate the displacement of curves from one frame to the other. Curves from one group are moved to fit the location of its matching group with corners serving as anchors. The best fit is then used to estimate the flow of the curve. The detection of occlusion or disocclusion, and the estimation of 2-D velocity are our current research topics.



(a) Fifth frame

(b) Fifteenth frame

Figure 9: Two frames from an image sequece with a chair



(a) Fifth frame

(b) Fifteenth frame

Figure 10: Matches among groups of smooth curves

4.5 INTEGRATED MOTION SYSTEM

In previous workshops[31], we have described the basic integrated motion system. We have continued to develop and use this system for testing each of the components (feature estraction, matching, motion estimation, and motion feedback to matching). This past year, we have concentrated on both the integration of the contour-based matching system with the region-based system, and the computation of three-dimensional depth and structure from the motion analysis. Three-dimensional motion estimation produces scaled estmates of the actual motion. The scale factor cannot be determined without other information (actual depth to a point, actual motion of a feature, etc.), but the relative depths can be derived from the different magnitudes of computed 3-D motions of features that are known to have the same actual 3-D motion (e.g. on the same object).

Region based approaches give a single motion and position estimate for all the points in the region. We are extending the capabilities of the system to produce more structure estimates for each matching region by using the contour based matcher to get correspondences for many points along the region contour. The contour matching programs assume that the positions of the contours are similar from frame to frame to reduce the searching necessary to produce a consistent match, but with the limited possible matches available when matching the contours from know matching regions (both an initial 2-D motion estimate and the lack of other possible matches) we can use the contour matcher with much larger disparities. The larger motion disparities are necessary to reduce the errors in the motion estimation process and to improve the structure estimates of the objects.

5 PARALLEL PROCESSING

We have several projects concerned with parallel processing techniques. These include one that is implementing current algorithms on the Connection Machine and another that is studying general techniques for implementing image understanding algorithms on parallel architectures.

Physical boundaries of objects are very important descriptors and are likely to generate edges during the imaging process. Even though the reverse is not true, it is reasonable to assume that the early stages in image analysis consist of detecting such discontinuities. Due to the complexity of the physical world and of the imaging apparatus, and to multiple sources of noise, the signal to be processed is complex, and the detection of such discontinuities is non trivial. Features detected locally are validated only by considering a more global context.

Computer vision problems always pose a challenge to today's computer systems. Early vision tasks occuring at the pixel level usually involve at least 64k (256×256) elements. Massively parallel processors can alleviate the problems of image processing because the operations are mostly spatially homogeneous. The Connection Machine [32] is a Single Instruction Multiple Data (SIMD) machine having between 16k and 64k processors. The Connection Machine Model CM-2 has 64 kilobits of bit-addressable memory for each processor instead of 4 kilobits for the CM-1. One of the two modes of communication among the processors is through the physical grid connection (the so-called NEWS network since the connections are in the four cardinal directions), allowing fast direct communication between neighboring processors. This facilitates the image processing, especially the low level processing of vision tasks.

To allow the machine to handle images with size more than 64k (or 16k), the Connection Machine supports virtual processors. A single physical processor can be divided into several virtual processors by serializing operations in time, and partitioning the memory in each processor. This allows the user to process images with sizes greater than the physical number of processors. As the virtual-to-physical (VP) ratio increases, the size of the local memory of each (virtual) processor decreases accordingly, and the speed of execution is slower than the speed of a physical processor by approximately the VP ratio.

Adaptive Smoothing is an edge preserving image smoothing algorithm in which we iteratively convolve the image with a mask whose coefficients reflect the degree of continuity of the underlying image surface. It has the nice property of detecting edges accurately at different scales. Higher level vision tasks, such as stereo and motion correspondence problems, can therefore be easily tackled by using a multiple scale approach with adaptive smoothing. Since the computation of Adaptive Smoothing requires only very local information, only a 3 by 3 neighborhood, it provides a direct massively parallel computation structure which is extremely suitable for the NEWS network on the Connection Machine. We have implemented it on the Connection Machine with 16k processors and 64k bits local memory per processor.

With the 16k processors, a $128 \times 128 \times 8bit$ image can be processed at one pixel per physical processor, namely the VP ratio is 1. We have experimented with the adaptive smoothing on images of various sizes with different VP ratio and observed its performance. Since the computation of adaptive smoothing involves the exponential function, the adaptive smoothing on the Connection Machine is performed using floating point arithmetic. With a Vax front end and the parallel Lisp (*Lisp) implementation, each iteration takes about 50 msecs on a 256 × 256 × *8bit* image without a floating-point accelerator. In order to compare the performance to the algorithm on a serial machine, we have also implemented the Adaptive Smoothing on a Symbolics 3645. It takes about 40 seconds for each iteration of adaptive smoothing on the Symbolics 3645 over an image of the same size (256 × 256 × *8bits*). Thus the speedup we get from the Connection Machine over the serial implementation is about three orders of magnitude.

To identify corresponding locations between two stereo images, or among a sequence of images in a motion analysis problem, is difficult because of the *false targets problem* (see [33]). The false targets problem can be alleviated either by reducing the range and resolution of the disparity or reducing the density of the matching features in the image. One commonly used method to obtain both resolution and range of disparity information is to apply a multi-resolution algorithm. At coarse resolution, the density of the matching features is low, therefore reducing the probability of false targets. The information obtained from the matching at coarse resolution can be used to guide the matching at fine resolution to get the desired high density disparity information.

Gaussian filtering has long been recognized as a popular smoothing operation. The scaling behavior prevents new features from appearing as the scale goes from fine to coarse. It and its derivatives can be efficiently implemented. Also it can be easily mathematically analyzed. Its accuracy on edge detection, however, has also been long criticized. When using Gaussian filtering as a multi-resolution operation for stereo matching, we have to deal not only with the tedious coarse-to-fine tracking along scale space but also with accuracy of the disparity information at coarse scale.

One of the most attracting features of our scheme is that the edge locations do not move along the scale, which enable us to construct a straightforward implementation of a multi-scale stereo matching algorithm. We have implemented a multiple scale stereo matching algorithm which is based on Drumheller and Poggio's [34] parallel stereo implementation on the Connection Machine. Our implementation not only greatly reduces the number of possible matches at each scale but also obtains a dense disparity map at fine scale.

In our work on parallel techniques for image understanding we have studied several storage and data access problems arising in mapping image algorithms onto parallel machines, parallel implementations of techniques developed by our group on hypercube and mesh based architectures, and continued our efforts in parallel computations on reconfigurable VLSI arrays and reduced meshes[35, 36]. (This work has been partially supported by AFOSR under grant AFOSR-89-0032.)

In iconic processing of image arrays several data storage and access problems arise. These problems become particularly important while implementing such tchniques on parallel machines. An image can be represented by a two dimensional array. Access to row vectors, column vectors, diagonals and subarrays are required heavily in image computations. Also, while implementing partitioned VLSI arrays and special purpose arrays, access to various subarrays is needed. We have developed a novel memory system for image processing. Latin squares, which are well known combinatorial objects for centuries, are used as the skew function of the memory system. We have introduced a new Latin square with desired properties for image array access. The resulting memory system provides access to various subsets of image data (rows, columns, diagonals, subarrays, etc.) in constant time while it uses a simple circuitry for address generation. This memory system is the first known memory system that achieves constant time access to rows, columns, diagonals and subarrays using minimum number of memory modules [37].

We have studied efficient parallel implementation of symbolic techniques developed by our vision group on hypercube SIMD arrays such as the Connection Machine. In particular, we have studied data movement technqiues for implementing stereo and image matching using high level primitives. By preprocessing the model, routing information is derived which is employed during the match phase. Our technique is simple and efficient and can be used on current parallel machines such as the Connection Machine. Notice that methods based on sorting can solve the data transport problems arising in the computation. However, in a model with N objects, these data transport problems can be solved in $O(\log N)$ time with a small constant factor by preprocessing the structure of the model. Similar techniques have been developed for performing such computations on mesh based architectures. We are currently implementing such data movement techniques on the Connection Machine at USC-ISI.

6 ACKNOWLEDGEMENTS

This paper describes the work of many faculty, staff and students. Besides contributing to research, they also provided pieces of text that are incorporated in this paper in some form or the other. In particular, we wish to acknowledge the contributions of Dr. V.K. Prasanna-Kumar, P. Saint-Marc, A. Huertas, W. Cole, R. Mohan, K. Rao, C. Peng, S. Cochran, F. Ulupinar, S. Gazit, and J. Chen.

References

[1] S. Cochran and G. Medioni. Accurate surface description from binocular stereo. In *Proceedings of the DARPA Image Understanding Workshop*, Palo Alto, California, May 1989.

[2] S. Gazit and G. Medioni. Multi-scale contour matching in a motion sequence. In *Proceedings of the DARPA Image Understanding Workshop*, Palo Alto, California, May 1989.

[3] Kashipati Rao and Ramakant Nevatia. Description of complex objects from incomplete and imperfect data. In *Proceedings of the DARPA Image Understanding Workshop*, Palo Alto, California, May 1989.

[4] R. Mohan and R. Nevatia. Perceptual organization for segmentation and description. In *Proceedings of the DARPA Image Understanding Workshop*, Palo Alto, California, May 1989.

[5] Jean-Luc Jezouin, Philippe Saint-Marc, and Gerard Medioni. Building an accurate range finder with off the shelf components. In *Proceedings of Computer Vision and Pattern Recognition Conference*, Ann Arbor, Michigan, June 1988.

[6] K. Sato and S. Inokuchi. Range-imaging system utilizing nematic liquid crystal mask. In *Proceedings of International Conference on Computer Vision*, pages 657–661, June 1987.

[7] T.J. Fan, G. Medioni, and R. Nevatia. Segmented descriptions of 3-d surfaces. *IEEE Journal of Robotics and Automation*, RA-3(6):527–538, December 1987.

[8] Philippe Saint-Marc and Gerard Medioni. Adaptive smoothing for feature extraction. In *Proceedings of the DARPA Image Understanding Workshop*, pages 1100–1113, Boston, Massachusetts, April 1988.

[9] P. Saint-Marc, J. Chen, and G. Medioni. Adaptive smoothing: A general tool for early vision. In *Proceedings of Computer Vision and Pattern Recognition Conference*, San Diego, California, June 1989.

[10] Ting-Jun Fan. *Describing and Recognizing 3-D Objects Using Surface Properties.* PhD thesis, University of Southern California, August 1988. Report USC IRIS 237.

[11] T.J. Fan, G. Medioni, and R. Nevatia. Recognizing 3-d objects using surface descriptions. In *2nd International Conference on Computer Vision*, pages 474–481, Tampa, Florida, December 1988.

[12] Kashipati G. Rao and R. Nevatia. Computing volume descriptions from sparse 3-d data. *International Journal of Computer Vision*, 2(1):33–50, June 1987.

[13] Gerard Medioni and R. Nevatia. Matching images using linear features. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 6(6):675–685, November 1984.

[14] R. Mohan, G. Medioni, and R. Nevatia. Stereo error detection, correction, and evaluation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 11(2):113–120, February 1989.

[15] K. A. Stevens. The visual interpretations of surface contours. *Artificial Intelligence*, 17:47–73, 1981.

[16] S. Tsuji and G. Xu. Inferring surfaces from boundaries. In *Proceedings of the 1st ICCV*, pages 716–720, 1987. London.

[17] R. Horaud and M. Brady. On the geometric interpretation of image contours. *Artificial Intelligence*, 37:333–353, 1988.

[18] Fatih Ulupinar and Ramakant Nevatia. Using symmetries for analysis of shape from contour. In *International Conference on Computer Vision*, Tampa, Florida, December 1988.

[19] Fatih Ulupinar and Ramakant Nevatia. Constraints for interpretation of perspective images. In *Proceedings of the DARPA Image Understanding Workshop*, Palo Alto, California, May 1989.

[20] A. Huertas and R. Nevatia. Detecting buildings in aerial images. *Computer Vision, Graphics and Image Processing*, 41(2):131–152, February 1988.

[21] R. Mohan and R. Nevatia. Perceptual grouping with applications to 3d shape extraction. In *IEEE Workshop on Computer Vision*, Miami, Florida, December 1987.

[22] A. Huertas, B. Cole, and R. Nevatia. Detecting runways in aerial images. In *Proceedings of the DARPA Image Understanding Workshop*, pages 272–297, Los Angeles, California, February 1987.

[23] FAA Advisory Circular. Marking paved areas on airports. Technical Report No. AC 150/5340-1E, Federal Aviation Administration, 1980.

[24] N. Ashford and P.H. Wright. *Airport Engineering, 2nd ed.* Wiley and Sons, 1984.

[25] A. Huertas, W. Cole, and R. Nevatia. Using generic knowledge in analysis of aerial scenes: A case study. in preparation, 1989.

[26] S.L. Gazit and G. Medioni. Contour correspondences in dynamic imagery. In *Proceedings of the DARPA Image Understanding Workshop*, pages 423–432, Boston, Massachusetts, April 1988.

[27] H. Shariat and K. Price. Motion estimation with more than two frames. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, to appear, 1989.

[28] Wolfgang O. Franzen. Representing motion using chronogeneous transformations. In *IEEE Workshop on Motion*, Newport Beach, California, March 1989.

[29] Shou-Ling Peng and Gerard Medioni. Spatio-temporal analysis of an image sequence with occlusion. In *Proceedings of the DARPA Image Understanding Workshop*, pages 433–442, Boston, Massachusetts, April 1988.

[30] Shou-Ling Peng and Gerard Medioni. Spatio-temporal analysis of velocity estimation of contours inan image sequence with occlusion. In *Proceedings of the International Conference on Pattern Recognition*, pages 236–241, Rome, Italy, November 1988.

[31] Keith Price and Igor Pavlin. Integration effort in knowledge-based vision techniques for the autonomous land vehicle program. In *Proceedings of the DARPA Image Understanding Workshop*, pages 417–421, Boston, Massachusetts, April 1988.

[32] D. Hillis. *The Connection Machine*. MIT Press, Cambridge, 1985.

[33] W.E.L. Grimson. *From Images to Surfaces: A Computational Study of the Human Early Visual System*. MIT Press, 1981.

[34] M. Drumheller and T. Poggio. On parallel stereo. In *Proceedings of the IEEE Conference on Robotics and Automation*, pages 527–538, December 1987.

[35] H. Alnuweiri and V.K. Prasanna-Kumar. Optimal image computations on reduced vlsi arrays. *IEEE Trans. on Circuits and Systems*, 1989.

[36] H. Alnuweiri and V.K. Prasanna-Kumar. Fast image labelling using local operators on mesh connected computers. In *International Conference on Parallel Processing*, 1989.

[37] Kichul Kim and V.K. Prasanna-Kumar. Parallel memory systems for image processing. In *Proceedings of Computer Vision and Pattern Recognition Conference*, San Diego, CA, June 1989.

IMAGE UNDERSTANDING RESEARCH AT THE UNIVERSITY OF MARYLAND (January 1988 – February 1989)

Azriel Rosenfeld
Larry S. Davis
John (Yiannis) Aloimonos

Computer Vision Laboratory
Center for Automation Research
University of Maryland
College Park, MD 20742-3411

ABSTRACT

This paper briefly summarizes research in image understanding conducted at the University of Maryland during the 14-month period January 1988 through February 1989. The areas covered include motion analysis, navigation, early vision, matching, parallel algorithms, pyramid techniques, and geometry.

MOTION ANALYSIS

CORRESPONDENCELESS METHODS

We have shown that a binocular observer can recover the depth and three-dimensional motion of a rigid planar patch, without using any correspondences between the left and right image frames (static) or between the successive image frames (dynamic). We have studied uniqueness and robustness issues with respect to this problem and have obtained experimental results from the application of our theory to synthetic and real images. [1]

SPACE-TIME FILTERING

In the process of extracting the optical flow through space-time filtering, we have to take into account constraints associated with the motion uncertainty, as well as with the spatial and temporal sampling rates of the temporal sequence of images. The motion uncertainty is shown to satisfy an inequality, as a consequence of the use of the Cramér-Rao inequality, which is a function of the filter parameters. On the other hand, the spatial and temporal sampling rates have lower bounds, which depend on the motion uncertainty, the maximum support in the frequency domain and the estimated optical flow. These lower bounds on the sampling rates and on the motion uncertainty are constraints which constitute an intrinsic part of the computational structure of space-time filtering. They are of a different nature than the ones used in regularization theory, because they do not dictate any arbitrary constraints on the parameters being computed, but instead arise as a natural consequence of the estimation process. By conjugating these constraints, we are able to devise an algorithm which describes an adaptive procedure of estimating the various parameters involved in space-time filtering. This corresponds to an instance of an adaptive system, through which the variables involved in the process of space-time filtering are allowed to vary inside a range which is consistent with the various intrinsic constraints governing the process. [2]

BINOCULAR FLOW ANALYSIS

We have analyzed image flow fields from parallel stereo cameras to determine the relative three-dimensional translational velocities of the camera platform with respect to objects in view and to establish stereo correspondence of features in the left and right images. A two step procedure is used. In the first step, the three components of the translational velocity are determined from linear equations whose coefficients consist of the sums of measured quantities in the two images. Separate equations are developed for cases where measurements of either the full optical flow or the normal flow are available. This computation does not require feature-to-feature correspondence. In the second step of the calculation, the same equations are used, with the computed translational velocities, as a constraint to find features in one image that correspond to given features in the other image. Preliminary experiments with synthetic flow fields indicate that the method gives accurate results even in the presence of noise. [3]

FLOW FROM TEMPORAL EDGES

A new method for the detection of motion and the computation of optical flow has been developed. In the first step of the calculation the intensity history at each pixel is convolved with the second derivative in time of a temporal Gaussian smoothing function. The zero crossings in a single frame of the resulting function indicate the positions of moving edges. Spatial and temporal derivatives of the function at the zero-crossing locations are then used to compute the component of the flow that is normal to the zero-crossing contours. Both the detection of motion and the computation of the normal velocity are insensitive to slow temporal and spatial changes in the image intensity that are caused by illumination effects rather than motion. The relationship of this work to gradient based flow measurement techniques has also been formulated. [4]

OPTIMAL USE OF POINT CORRESPONDENCES

One of the problems associated with any approach to the structure from motion problem using point correspondences, i.e. recovering the structure of a moving object from its successive images, is the use of least squares on dependent variables. We have formulated the problem as a quadratic minimization problem with a non-linear constraint. We have derived the condition for the solution to be optimal under the assumption of Gaussian noise in the input, in the Maximum Likelihood Principle sense. This constraint minimization reduces to the solution of a non-linear system which in the presence of modest noise is easy to approximate. We have developed two efficient ways to approximate it, and have defined some inherent limitations of the structure from motion problem when two frames are used that should be taken into account in robotics applications that involve dynamic imagery. Our formulation introduces a framework in which previous results on the subject become special cases. [5]

TRACKING

A mathematical theory for visual tracking of a three-dimensional target of known shape moving rigidly in 3-D has been developed, and it has been shown how a monocular observer can track an initially foveated object and keep it stationary in the center of the visual field. Our goal is to develop correspondence-free tracking schemes and get rid of the limitations inherent in the optical flow formalism. A general tracking criterion, the *Tracking Constraint*, has been derived, which reduces tracking to an appropriate optimization problem. The connection of our tracking strategies with the Active Vision Paradigm has been shown to provide a solution to the Egomotion problem under the assumption of knowledge of shape. Tracking strategies based on the recovery of the 3-D motion of the target have been devised under the above assumption. A correspondence-free scheme has been derived, which depends on global information about the scene (provided by linear features of the image) in order to bypass the ill-posed problem of computing the spatial derivatives of the image intensity function, and amounts to the solution of a linear system of equations in order to estimate the 3-D motion of the target. An important feature of these tracking strategies is that they do not require continuous segmentation of the image in order to locate the target. Supposing that the target is sufficiently textured, dynamic segmentation using temporal derivatives of the linear features provides sufficient information for the tracking phase. Therefore, this approach should perform best when previous ones fail, namely in a complex visual environment. Experimental results demonstrate robustness in the presence of noise. [6]

APPARENT MOTION

The existence of two separate mechanisms for the processing of apparent motion, the short- and long-range processes, as proposed by Braddick in 1974, has been analyzed through many different psychophysical experiments. In particular the fact that for the short-range process there exists an upper bound for the spatial displacement and temporal interstimulus interval between successive stimulus presentations was confirmed by several of these experiments. In order to gain a more formal understanding of these issues, we have analyzed the phenomenon of apparent motion from the point of view of a reconstruction problem. This allowed us to use the sampling theorem to analyze the problem of temporal (spatial) reconstruction of uniformly translating patterns. In the case where the velocity field can only be extracted with uncertainty, it can be shown that there exists a maximum temporal (spatial) sampling interval, such that aliasing does not occur. We suggest that, in the case of the short-range process, due to its temporal (spatial) reconstruction ability, a similar effect could intervene in the limitation of its activity to a small spatio-temporal scale. [7]

ERROR ANALYSIS

Relative motion between objects and the viewer generates a time-varying image which, in principle, can be used as a source of 3D information about the structure of the objects and the relative motion. One approach to obtaining 3D information from time-varying imagery is to utilize the image flow field and its derivatives. The characteristics of the image flow field depend both on the relative motion and the surface of the object. Thus, given the image flow field, in theory, one can invert the problem and recover the relative motion and the structure of the object. We have analyzed the intrinsic reliability of such an approach; i.e. assuming that the image flow field is known accurately, except for quantization error, we have derived closed-form expressions for the error due to quantization in the recovered *3D motion and structure parameters. These expressions are essential* for revealing the intrinsic limitations of the approaches used for the recovery of the 3D parameters from a given image flow field and are thus of great practical importance. [8]

MULTI-FRAME METHODS

The main issue in the area of motion estimation given the correspondences of some features in a sequence of images is sensitivity to error in the input. The main way to attack the problem is redundancy in the data. Up to now all the algorithms developed either used two frames or depended on restrictive assumptions and ad hoc techniques. We have developed an algorithm based on multiple frames that employs only the rigidity assumption, is simple and mathematically elegant, extremely flexible and, most importantly, is a major improvement over the two-frame algorithms. The algorithm does minimization of the mean square error, which we have proved equivalent to an eigenvalue minimization problem. One of the side effects of this mean square method is that the algorithm has a very descriptive physical interpretation in terms of the "loaded spring model". [9]

NAVIGATION

ROAD FOLLOWING

A method for the reconstruction of a road in 3D space from a single image has been developed. The world road is modelled as a space ribbon generated by a centerline spine and horizontal *cross-segments* of constant length (the *road width*) cutting the spine at their midpoints and normal to the spine. The tangents to the road edges at the end points of cross-segments are also assumed to be approximately parallel. These added constraints are used to find pairs of points (*matching points*) which are images of the end points of world cross-segments. Given a point on one road image edge, the method finds the matching point(s) on the other road image edge. For images of road turns, a point on one road image edge has generally more than one matching point on the other edge. The extra points belong to "ghost roads" whose images are tangent to the given road image at these matching points.

Once pairs of matching points are found in the image, the reconstruction of the corresponding world cross-segments is straightforward since cross-segments are assumed to be horizontal and to have a known length. Ghost road cross-segments are discarded by a dynamic programming technique. A benchmark using synthetic roads has been used in tests of the method, and the sensitivity of the road reconstruction to variations in width and bank of the actual world road has been evaluated and compared to the sensitivity of two other algorithms. Experiments with a sequence of actual road images as the Autonomous Land Vehicle (ALV) moves down a road have also been performed. [10]

A new scheme for reconstructing the 3D shape of roads from camera images was subsequently developed based on the *local flatness approximation.* In this scheme, all equations are written in terms of *NHC vectors* defined by quantities directly observable on the image plane. Hence, analysis is done solely in the image domain: No 3D solution is constructed in the scene. Much consideration was given to computational stability with regard to possible inaccuracy of image data. A relaxation scheme was defined which always guarantees the global consistency of the computed solution. The singularities of the constraint resulting from the local flatness approximation has also been analyzed. [11]

MOTION PLANNING

Motion planning for a point robot has been studied in a time-varying environment. Obstacles are convex polygons which move in a fixed direction at a constant speed. The point to be reached (referred to as the destination point) also moves along a known path. The concept of "accessibility" from a point to a moving object is introduced, and is used to define a graph on a set of moving obstacles. The graph is shown to exhibit an important property: if the moving point is able to move faster than any of the obstacles, a time-minimal path is given as a sequence of edges in the graph. An algorithm has been developed for generating a time-minimal path and its execution time has been analyzed. [12]

QUALITATIVE NAVIGATION

Visual navigation is a major goal in machine vision research, and one of both practical and basic scientific significance. The practical interest reflects a desire to produce systems which move about the world with some degree of autonomy. The scientific interest arises from the fact that navigation seems to be one of the primary functions of vision in biological systems. Navigation has typically been approached through reconstructive techniques since a quantitative description of the environment allows well understood geometric principles to be used to determine a course. However, reconstructive vision has had limited success in extracting accurate information from real-world images. We have shown that a number of basic navigational operations can be realized using qualitative methods based on inexact measurement and pattern recognition techniques.

Navigational capabilities form a natural hierarchy beginning with simple abilities such as orientation and obstacle avoidance, and extending to more complex ones such as target pursuit and homing. Within a system, the levels can operate more or less independently, with only occasional interaction necessary. We have studied three basic navigational abilities: *passive navigation*, *obstacle avoidance*, and *visual homing*, which together represent a solid set of elementary, navigational tools for practical applications. It has been demonstrated that all three can be approached by qualitative, pattern-recognition techniques. For passive navigation, global patterns in the spherical motion field can be used to robustly determine the motion parameters. For obstacle avoidance, divergence-like measurements on the motion field can be used to warn of potential collisions. For visual homing an associative memory can be used to construct a system which can be trained to home visually in a wide variety of natural environments. Theoretical analyses of the techniques have been presented, and working systems have been implemented and tested. [13]

EARLY VISION

BOUNDARY-PRESERVING REGULARIZATION

Many problems in low-level vision and in several other scientific or engineering disciplines are ill-posed in the sense that their solutions do not exist, are not unique, or do not depend continuously on the data. We approach these problems with Tikhonov regularization. That means we seek a solution that is a compromise between the requirements of consistency with constraints imposed by the data and of consistency with a priori smoothness assumptions. Unfortunately, the solution obtained blurs boundaries and makes it hard to recognize where the real world variables change sharply. We approach this difficulty by assuming the errors (the inconsistency between data and solution) at nearby points are correlated and we first deblur the errors before regularizing. Similarly we have to deblur the smoothness term of our variational condition before we can apply regularization theory. In general decorrelation is a hard problem, but making special assumptions about the blurring kernel (e.g. the kernel is Gaussian or more generally Levy stable), we can recover the magnitude of the deblurred error (or smoothness) as a linear expression in terms of the original error (or smoothness) and its derivatives. We are, in effect, imposing a requirement that not only the error but also its derivatives should tend to be small (because noise is often far from being white). The resulting variational condition is not the optimal condition but the Euler-Lagrange equations will be linear if the constraints are. We also suggest a convex approximation technique for solving the piece-wise smooth interpolation problem which results in a convex condition if the original constraints were linear. [14]

SIGNAL AND NOISE ESTIMATION

When we examine a set of data, it is often "obvious" that the data can be interpreted as values of a particular type of function (e.g., linear) corrupted by a particular type of noise (e.g., zero-mean, spatially stationary). We have investigated a qualitative approach, based on Bayes' theorem, that may justify such interpretations. We have dealt primarily with data that are samples of a real-valued function of a single variable, but similar ideas apply to functions of two or more variables, to vector-valued functions (e.g., curves or surfaces), as well as to the problem of finding natural clusters in sets of points. [15]

An algorithm has been developed and tested for estimating noise variance in images. The only information available to the algorithm is the corrupted image and the white nature of the zero mean Gaussian noise. The algorithm recovers the variance of the noise in two steps. First, the sample variances are computed for square cells tessellating the noisy image. Several tessellations are applied with the size of the cells increasing four-fold for consecutive tessellations. The four smallest sample variance values (the outcomes of the first four order statistics) are retained for each tessellation and combined through an outlier analysis into one estimate. The different tessellations thus yield a variance estimate sequence. In the second part of the algorithm, the value of the noise variance is determined from this variance estimate sequence. The algorithm has been applied to 500 noisy 256×256 images derived from seven prototypes of classes often employed in computer vision and image processing. In 98% of the cases the relative estimation error was less than 0.2 with an average error of 0.06. All the operations in the algorithm are parallel and if they are implemented on an image pyramid, the variance of the noise is recovered in O[log(image_size)] processing time. [16]

CLUSTER DETECTION IN NOISE

If a feature space contains a set of clusters and background noise, it may be difficult to extract the clusters correctly. In particular, when we use a partitioning scheme such as k-means clustering, where k is the correct number of clusters, the background noise points are forced to join the clusters, thus biasing their statistics. We have developed a preprocessing technique that gives each data point a weight related to the density of data points in its vicinity. Points belonging to clusters thus get relatively high weights, while background noise points get relatively low weights. k-means clustering of the resulting weighted points converges faster and yields more accurate clusters. [17]

Cluster detection algorithms using mean values as center estimates suffer from inaccuracy when the distributions of points in clusters are not Gaussian or when unevenly distributed background noise is present. We have developed a mode-based cluster detection algorithm using a least median square error measure for the center estimates. The algorithm locates the centers with reasonable accuracy under biased background noise. Its complexity is $O(n \log n)$ in general, and this is reduced to $O(n)$ for data on a lattice. [18]

DIFFERENTIATION

Computation of the derivatives of an image defined on a lattice structure is of paramount importance in computer vision. The solution implies least square fitting of a continuous function to a neighborhood centered on the site where the value of the derivative is sought. We have developed a systematic approach to the problem involving orthonormal bases spanning the vector space defined over the neighborhood. Derivatives of any order can be obtained by convolving the image with a priori known filters. We have shown that if orthonormal polynomial bases are employed the filters have closed form solutions. The same filter is obtained when the fitted polynomial functions have one consecutive degree. Moment preserving properties, sparse structure for some of the filters, and relationship to the Marr-Hildreth and Canny edge detectors have also been established. Expressions for the filters corresponding to fitting polynomials up to degree six and differentiation orders up to five, for the cases of unweighted data and data weighted by the discrete approximation of a Gaussian, have been tabulated. [19]

LINE FITTING

A method to improve the estimate of least squares line fits to thin stripes in images has been developed. By using the geometry of local gray level patterns and their contrasts, the accuracy of the least squares line fits can be improved markedly. The improved method's performance is comparable to that of the Canny line detector. [20]

In fitting a straight line to a noisy image, the least square method becomes unreliable if non-Gaussian outliers are present. We have developed a the Least Median Square (LMS) method, which provides:

- protection against distortion by up to 50% of contaminated data;
- good efficiency in the presence of various type of noise;
- an amount of computation comparable with the least square method. [21]

HYPERACUITY

In spatial hyperacuity the subjects discriminate a stimulus feature relative to a reference, with an accuracy significantly better than the grain of the retinal mosaic. We have shown that the normalized thresholds have a dichotomous behavior; they are either insensitive to the spatial parameter in the experiment or increase very steeply with it. This behavior is explained by the involvement in the processing of pixel (receptor) accuracy information about the structure of the stimulus. A computational model employing optimal filtering reproduces the experimental data and suggests that processing of spatial hyperacuity tasks in the human visual system is optimal. [22]

TEXTURE SEGREGATION

We have studied human perception of texture segregation in patterns composed of two textures where each texture contained two types of elements. The elements were arranged in a striped pattern in the top and bottom regions and in a checked pattern in the center region. The observers rated the degree to which the three regions were seen as distinct. When the elements were squares or lines, perceived segregation resulting from differences in element size could be cancelled by differences in element contrast. Minimal perceived segregation occurred when the products of the area and contrast (areal contrasts) of the elements were equal. This dependence of perceived segregation on the areal contrasts of the elements is consistent with a simple model based on the hypothesis that the perceived segregation of the regions is a function of their differential stimulation of spatial frequency channels. However, two aspects of the data were not consistent with quantitative predictions of the model. First, as the size difference between the large and small elements increased, the ratings at the point of minimum perceived segregation increased. Second, the effect of changing the fundamental frequency of the textures was not predicted by the model. These discrepancies may be explained by a more complex model in which a rectification or similar nonlinearity occurs between two stages of orientation- and spatial-frequency-selective linear filters. [23]

LOCAL OPERATIONS ON DOT PATTERNS

When a local operation is performed on the pixels in an array, the new value of the pixel is a function of the old values of the pixel and its neighbors. We have introduced the more general concept of local operations on labelled dot patterns, where the new label of a dot is a function of the old labels of the dot and a set of its neighbors (e.g., its Voronoi neighbors). Such operations may change the positions of the dots, in addition to changing their "values". These ideas are illustrated with examples of operations that perform local feature detection (e.g., isolated dot detection, cluster edge detection, dotted curve detection) and "enhancement" (e.g., "smoothing" the dot spacing or "sharpening" the edges of diffuse clusters), as well as "morphological" operations. [24]

MATCHING

ORDERED MATCHING

Matching of two digital images is computationally expensive, because it requires a pixel-by-pixel comparison of the pixels in the image and in the template. If we have probabilistic models for the classes of images being matched, we can reduce the expected computational cost of matching by comparing the pixels in an appropriate order. We have shown that the expected cumulative error when matching an image and a template is maximized by using an ordering technique. We have also presented experimental results for digital images, when we know the probability densities of their gray levels, or more generally, the probability densities of arrays of local property values derived from the images. [25]

A generalization of the ordered matching problem is the problem of optimally ordering a set of operations, the outcomes of which are random. We have developed procedures for finding the optimal dynamic strategy and the optimal static strategy for solving this problem. We have also considered a constrained form of the problem and shown that it has a simple optimal strategy, and we have investigated the complexity issues involved in finding optimal strategies. [26]

LOCATION SELECTION IN MATCHING

We have developed a technique to further reduce the computational cost of template matching by using probabilistic knowledge about local features that appear in the image and the template. Using this technique the most probable locations for successful matching can be found. We have analyzed how the size of the features affects the computational cost and the robustness of the technique. We have shown experimentally that even simple methods of feature extraction and representation can reduce the computational cost by more than an order of magnitude. [27]

MATCHING RUN LENGTH CODES

We have developed an algorithm to reduce the computational cost of template matching by using run length representation of the image and the template. Using this technique we compare only locations in the image and the template where the total mismatch accumulation may change. This method works best for images and templates with long runs. We have studied conditions under which the algorithm will be efficient, and tested it experimentally on both randomly generated and real images. In some cases, using this approach yields more than 20-fold speedup. [28]

MATCHING POLYGONAL ARCS

We have developed an efficient algorithm for matching two rectilinear polygonal arcs. We first show how to match two arcs of the same length by decomposing them into a set of pairs of corresponding straight line segments having the same length. The distance measure of each such pair of line segments is calculated by referring to the distance of one of six possible configurations of pairs of segments. We then show how to find the relative position of the two arcs which yields the best match by minimizing the distance function. After analyzing the case of arcs having the same length, we show how to use the results and a representation of rectilinear arcs as strings generated by four primitives to obtain an efficient algorithm for arc matching. This algorithm is based on our earlier algorithm for run-length string matching. [29]

BOOLEAN OPERATIONS ON POLYGONS

A robust algorithm for set operations on pairs of polygons has been developed. The algorithm is capable of operating on the class of vertex-complete polygons which properly includes the simple polygons. The algorithm uses carefully chosen data structures and is easy to describe. We have given a proof of its correctness and an analysis of its complexity. [30] It has also been generalized to sets of polygons, using a boundary representation for the input and output polygon sets. The polygons in each set can be either island or hole polygons. We show how to make the basic algorithm more efficient by using inclusion trees that can be built from the polygons in each set. The implementation is table-driven and is facilitated by the use of efficient data structures. The algorithm can be applied to efficient matching of images that can be decomposed into regions having polygonal boundaries. [31]

MAP REGISTRATION

To obtain a map of the ocean floor, a multibeam echo-sounder system capable of measuring depth is installed aboard a ship. The ship sails for several miles along a straight track and collects a swath of depth data. Then it changes its course and collects another swath of data, doing this repeatedly in such a way that each swath overlaps with a few others. However, because in the middle of the ocean it is very difficult for the ship to know its accurate position, the overlapping swaths are almost always misregistered with respect to each other. We have developed an automated system for obtaining a correctly registered map of the ocean floor. Because each swath of data overlaps with several others, the registration is performed both at local and global levels. The "primitives" used for local matching are contours of constant depth which are extracted from the data and are represented by means of a modified chain code method. The main heuristic guiding the search for matching contours of equal depth is their apparent proximity to the middle of the unregistered overlapping region. The degree to which two contours match is determined by the correlation of their respective chain codes and the geometrical proximity of their nodes. All "best" matches are considered tentative until their geometrical implications are evaluated and a consistent majority has emerged. To do global matching a cost function has been constructed and minimized. Terms contributing to the cost include violation of local matches as well as compression and bending of the swaths of data. [32]

SYMBOLIC MODEL MATCHING

Existing expert vision systems generally match models to images using only numeric "goodness-of-fit" measures. The computation of such measures usually involves the combining of incommensurate quantities and the loss of low level knowledge that could be useful at higher levels. The methods employed, and hence the software developed, often cannot be generalized for use within other domains or at other levels of abstraction. We feel that there is a need for a more general symbolic image/model matching paradigm, and for the development of software tools that implement it. We have formulated motivations for the development of a general purpose symbolic matcher, developed an implementation, tested it on real-world image lata, and discussed important requirements that any such system ought to meet. [33]

PARALLEL ALGORITHMS

GRAPH MATCHING

We have performed experiments with a parallel algorithm for matching attributed relational graphs. The algorithm generates a state space tree in a breadth-first manner and then evaluates the tree by computing the edit distance for each candidate solution. The parallelization method used is best suited for MIMD-type computers. The first target machine is the Butterfly Parallel Processor, in which the programs were developed on Uniform System software supporting a shared memory model of computation. The second multiprocessor is a link-oriented Transputer-based system. In this system, concurrent processes communicate through message channels. The experiments showed that nearly linear speedup can be achieved by parallelizing the algorithm in the outermost loop. [34]

BORDER TRACKING

We have studied parallel implementation of a generalized one-pass algorithm for border tracking of objects in thresholded binary images. The input image is scanned from top to bottom, from left to right. On each row, partial border descriptions produced on previous rows are updated according to run ends on the current row. Borders are represented by crack code strings following the outer borders in a clockwise direction, and the inner borders in a counterclockwise direction. The parallelized version of the algorithm has been implemented on a Butterfly Parallel Processor. The program was developed based on the Uniform System approach. The input image is partitioned into equal sized blocks, and each partition is assigned to a separate processor. Partially completed border descriptions gathered from the blocks are finally merged in parallel. [35]

POSE ESTIMATION

We have implemented Linnainmaa and Harwood's pose determination algorithm on the Butterfly Parallel Processor (BPP) and Hathi 2 parallel computers. The architecture of the BPP is based on a shared memory model, whereas the Hathi 2 is based on a distributed memory model. The algorithm is computationally very intensive, which makes it suitable for parallel processing. The program was parallelized using the processor farm technique, thus enabling automatic load balancing. The experiments show that the algorithm is very easy to parallelize. Furthermore comparison of the two architectures shows that the Hathi 2 is much more powerful than the BPP. Due to different implementation technologies, it is not, however, possible to say whether one of the architectures is in general better than the other. [36]

HIDDEN SURFACE COMPUTATION

We have developed a data parallel quad-tree algorithm for computing hidden edges in a scene consisting of polygons in 3-space. The algorithm is based on Warnock's hidden-edge algorithm, but actually computes a quad-tree representation of the image, rather than the image itself. It runs in time proportional to the number of polygons in the scene and to the log of the desired resolution. It has been implemented on the Connection Machine. [37]

GENERALIZED MATRIX INVERSION

The generalized inversion of a matrix has many applications. We have studied the parallel implementation of the Ben-Israel-Greville algorithm for finding the Moore-Penrose inverse of a matrix. This algorithm is highly suitable for data-level parallelism and has several advantages: linearity, stability, reliability, determinism and scalability. Connection Machine experiments with random matrices of different dimensions have been performed. [38]

Theoretical results concerning partitioning of large matrices for g-inversion have also been investigated, and the complexity and performance analysis of these methods on the Connection Machine have been studied. It turns out that the use of the virtual processor configuration on the Connection Machine is of comparable efficiency to using any partitioning scheme, when the multiplicative iterative scheme is used for g-inversion. [39]

TENSOR PRODUCTS

Tensor products are widely used in the evaluation and interpolation of functions as well as 2D and 3D image blocks. We have also implemented the tensor product method on the Connection Machine. [40]

SPLINE-BLENDING APPROXIMATION

Wee have studied the projection operator technique for multivariable cardinal spline-blending approximation on the Connection Machine. This technique requires data-parallel operations for polynomial (single and multivariable) evaluation and hence is well suited for implementation on the Connection Machine. The basic operations needed are the inner product and the tensor product of vectors whose components are polynomials or their evaluated values. Spline-blending approximation has several applications: finite-element methods, digital image processing, optical flow and topography. [41]

NEURAL NETWORK SIMULATION

Partitioning a set of N patterns in a d-dimensional metric space into K clusters—in a way that those in a given cluster are more similar to each other than the rest—is a problem of interest in image analysis, astrophysics and other fields. As there are approximately $\frac{K^N}{K!}$ possible ways of partitioning the patterns among K clusters, finding the best solution is beyond exhaustive search when N is large. We have shown that this problem in spite of its exponential complexity can be formulated as an optimization problem for which very good, but not necessarily optimal, solutions can be found by using a neural network. To do this the network must start from many randomly selected initial states. The network has been simulated on the NASA MPP (a 128×128 SIMD array machine), where we used the massive parallelism not only in solving the differential equations that govern the evolution of the network, but also in starting the network from many initial states at once thus obtaining many *solutions in one run*. We have obtained speedups of two to three orders of magnitude over serial implementations. [42]

PYRAMID SIMULATION

We have developed an algorithm for fast addition on the fat pyramid. The fat pyramid is a pyramid in which the storage space and the processing power allocated to a single node increase as the root of the pyramid is approached. The addition algorithm is based on a carry-lookahead technique. The computation time of the algorithm is proportional to $\log p + q$ for operands of size $p * q$ bits, when p processors are used to deal with the numbers. The addition algorithm was simulated on the Connection Machine. [43]

A pyramid programming environment on the Connection Machine has been developed. The mapping between the Connection Machine and pyramid structures is based on a scheme called Shuffled 2D Gray Codes. A pyramid Hough transform, based on computing the distances between line or edge segments and enforcing merge and select strategies among them, has been implemented using this programming environment. [44]

PYRAMID TECHNIQUES

PYRAMIDS AND PRISMS

An image pyramid is a hierarchy of representations of the input derived by recursive smoothing and decimation. Image pyramids are built in log(image_size) time with the consecutive levels having their size and resolution reduced by a constant factor. Similar structures with the representations decreasing only in resolution but not in size are also of interest. Such constant size multiresolution representations of the input can be simulated on image pyramids by increasing the number of values stored in the cells of the host structure. Constant size representations allow parallel processing in applications such as *scale-space filtering and multiresolution* edge detection. [45]

PYRAMID ROBUSTNESS

Image pyramids have been used by many investigators as computational structures for multi-resolution image processing and analysis. We have subjected such pyramids to various structural perturbations and investigated their effects on the functions of the pyramid. The perturbations ranged from adding Gaussian noise to the weights of the generating kernel, to generating a hierarchy of completely irregular tessellations of the image field. We have shown that homogeneous parts of the low resolution representations of the input image may be recovered by renormalizing the corrupted weights. Multi-resolution algorithms transposed to irregular (stochastic) structures exhibited only a small decrease in performance. We conclude that pyramidal algorithms are robust and are only weakly dependent on the underlying structure. We suggest that some of these pyramidal algorithms may also serve as computational models for perceptual phenomena. [46]

BIMODALITY ANALYSIS

The bimodality of a population P can be measured by dividing its range into two intervals so as to maximize the Fisher distance between the resulting two subpopulations P_1 and P_2. If P is a mixture of two (approximately) Gaussian subpopulations, then P_1 and P_2 are good approximations to the original Gaussians, if their Fisher distance is great enough. For a histogram having n bins this method of bimodality analysis requires $n-1$ Fisher distance computations, since the range can be divided into two intervals in $n-1$ ways. The method can also be applied to "circular" histograms, e.g. of populations of slope or hue values; but for such histograms it is much more computationally costly, since a circular histogram having n bins can be divided into two intervals (arcs) in $n(n-1)/2$ ways. The cost can be reduced by performing bimodality analysis on a "reduced-resolution" histogram having n/k bins; finding the subdivision of this histogram that maximizes the Fisher distance; and then finding a maximum Fisher distance subdivision of the full-resolution histogram in the neighborhood of this subdivision. This reduces the required number of Fisher distance computations to $n(n-1)/2k^2+O(k)$. For histograms representing mixtures of two Gaussians, this method was found to work well for n/k as small as 8. [47]

IMAGE SEGMENTATION

If an image contains regions whose gray level populations differ only slightly from that of the background, it may be difficult to detect their presence by statistical population analysis, since they may not give rise to significant bimodality. If the regions are relatively compact, however, in the sense that they do not consist primarily of border pixels, the image's bimodality can be significantly increased by local averaging. Thus local averaging followed by bimodality analysis can be used to detect compact regions that differ slightly in gray level population from their background. This method may also be useful in detecting objects that differ texturally from their backgrounds, where initial local filtering may yield only slight differences between the object and background gray level populations in the filtered images. [48]

BORDER DELINEATION

A pyramid technique for delineation of compact objects has been developed. The borders of the objects are detected in a low resolution representation of the input, a higher level of the pyramid. The pixels on the two sides of an edge are the roots for two classes (object and background). The two classes are employed in two independent top-down tree growing processes. The information is passed downward by adjusting confidence measures. The employment of multiple roots defined on the smoothed representation of the input contributes to the robustness of the method at very low signal-to-noise ratios. [49]

HOUGH TRANSFORM

We have developed a divide-and-conquer Hough transform technique for detecting a given number of straight edges or lines in an image. This technique is designed for implementation on a pyramid, and requires only $O(\log n)$ computational steps for an image of size $n \times n$. [50]

CONTOUR PROCESSING

A novel hierarchical approach toward fast parallel processing of chain-codable contours has been developed. The environment, called the chain pyramid, is similar to a regular non-overlapping image pyramid structure. The artifacts of contour processing on pyramids are eliminated by a probabilistic allocation algorithm. Building of the chain pyramid is modular, and for different applications new algorithms can be incorporated. We have implemented two applications: smoothing of multi-scale curves, and gap bridging in fragmented data. The latter is also employed for the treatment of branch points in the input contours. A preprocessing module allowing the application of the chain pyramid to raw edge data has also been developed. The chain pyramid makes possible fast, O[log(image_size)], computation of contour representations in discrete scale-space. [51]

PICTURE PARSING

We believe that pyramids are a natural architecture for implementing a general method of syntactic pattern recognition. Pyramids can be used to extract syntactic primitives (local features, edges/curves, or regions of simple shapes) from an image and to compute their properties They can also be used to identify hierarchical arrangements of primitives, thereby parsing the image (in parallel) in accordance with the rewriting rules of a "coordinate grammar". [52]

GEOMETRY

HEXAGONAL GRIDS

Square and hexagonal spatial samplings, because of their processing ease, are used most widely in image and signal processing. However, no rigorous treatment of the quantization error due to hexagonal sampling has appeared in the literature. We have developed mathematical tools for estimating quantization error in hexagonal sensory configurations. These include analytic expressions for the average error and the error distribution of a function of an arbitrarily large number of hexagonally quantized variables. The two quantities, the average error and the error distribution, are essential in assessing the reliability of a given algorithm. For comparison we have also computed the corresponding expressions for square spatial sampling, so that they can be used in comparing the magnitude of the error incurred in hexagonal versus square quantization for a given algorithm. They can thus be used to determine which sampling technique would result in less quantization error for a particular algorithm. Such a comparison is important due to the paramount role that quantization error plays in computational approaches to computer vision. [53]

METRICS

In computer vision a variety of metrics are used to determine the distance between lattice points. The three which are encountered most often are city block distance, chessboard distance, and Euclidean distance. A set S of lattice points will be said to be (r, s)-metrically independent if the congruence of S and T under the r metric implies congruence under the s metric for every digital set T. Necessary and sufficient conditions are obtained for sets to be metrically independent with respect to the three given distances. Conditions on the interpoint distances are also determined which permit a set to be imbedded in the digital plane with these metrics. [54]

DIGITAL GEOMETRY ON GRAPHS

Many of the standard concepts of digital geometry, particularly those involving connectedness and distance properties of subsets of a digital image, can be generalized to subgraphs of an arbitrary graph G. Algorithms for connected component labeling, distance transform computation, etc., can be defined that require time $O(n)$, where n is the number of nodes of G. Parallel algorithms for these computations can also be defined, using various modes of parallel computation. We can also define "continuous" integer-valued functions on graphs, and can show that the distance transform is the largest such function having (at least) a given set of zeros. [55]

CONTOUR CODES

An *isothetic polygonal arc* is one that has all its sides oriented in two orthogonal directions, so that all its angles are right angles. Such an arc is determined (up to congruence) by specifying a "code" sequence of the form $\alpha_1 A_1 \alpha_2 \cdots \alpha_{m-1} A_{m-1} \alpha_m$, where the α's are positive real numbers representing side lengths, and the A's are single bits that specify whether the arc turns left or right between one side and the next. We have developed basic properties of this code, and shown how to derive various geometric properties of the arc (or the region it bounds, if it is closed) directly from the code. [56]

MEDIAL AXIS TRANSFORMS

The Medial Axis Transform represents a region of a digital image as the union of maximal upright squares contained in the region. We have studied the problem of computing geometric properties of the image from a representation that generalizes the squares to rectangles. We have given algorithms for a number of problems using n processors where n is the number of upright rectangles. Our algorithms compute the perimeter, eccentricity, center of gravity, moment of inertia and area of the region covered by the rectangles in $O(\log n)$ time. The results are faster than previous results and are optimal (to within a constant factor). The *contour* of such a region may contain as much as $O(n^2)$ pieces; our algorithm computes the contour with a worst case running time of $O(n)$. We also give an optimal parallel algorithm to construct the medial axis transform representation given an array representation of the image. [57]

DOT PATTERNS

We have defined random processes that generate planar dot patterns in which the dots have a tendency to cluster, or in which clustering is inhibited. We have also defined processes for labeling a given point pattern in such a way that neighboring points tend to have, or not to have, the same labels. The patterns generated by such processes can serves as test data for image analysis algorithms that operate on spatial configurations of local image features. [58]

MARKOV RANDOM FIELDS

Generating a Markov random field image is a computationally very expensive process on a sequential processor. We have developed a parallel algorithm to perform this task and have implemented it on the Connection Machine. We show on theoretical and experimental grounds that a 40% degree of parallelism is optimal for this algorithm. In our implementation we demonstrate a 40% degree of parallelism and an effective speedup of more than 70 times over the sequential implementation on a Vax 11/785 running Unix. [59]

OTHER TOPICS

TESTING GEOMETRICAL CONFIGURATIONS

We have developed a general formulation for testing particular geometrical configurations of image data. The procedure consists of hypothesizing and testing: We first estimate an ideal geometrical configuration which supposedly exists, and then check to what extent the original edge data must be displaced to support the hypothesis. Thus, all types of tests are reduced to computing a single measure of edge displacement without involving ad-hoc measures and threshold values depending on the problem. Also, no explicit forms of probability distribution need be introduced. All the procedures are described by explicit algebraic expressions in unit vectors which represent points and lines on the image plane, so that no computational overflow occurs and no searches or iterations are required. [60]

IMAGE INTERPRETATION: PROGRAMMED PICTURE LOGIC

The objective of the PPL project is to design and implement a general and modular logic-programmed system for two-dimensional interpretation of image theories in image structures obtained by image analysis. Important subsystems include heuristic search for object instances with optimization of goodness-of-figure, and procedures for computing basic image components, locales for searches, and predicates. Some of these have been illustrated in an application to aerial images of suburban neighborhoods. [61]

PLANNING

The value of enabling a planning system to remember the plans it generates for later use was acknowledged early in planning research. The systems developed, however, were very inflexible as the reuse was primarily based on simple strategies of generalization via variablization and later unification. We have developed an approach for flexible reuse of old plans in the presence of a generative planner. In our approach the planner leaves information relevant to the reuse process in the form of annotations on every generated plan. To reuse an old plan in solving a new problem, the old plan along with its annotations is mapped into the new problem. A process of ***annotation verification*** is used to locate applicability failures and suggest refitting strategies. The planner is then called upon to carry out the suggested modifications—to produce an executable plan for the new problem. This integrated approach obviates the need for any extra domain knowledge (other than that already known to the planner) during reuse and thus affords a relatively domain-independent framework for plan reuse. We have studied the realization of this approach in two disparate domains (blocks world and process planning for automated manufacturing) and have proposed extensions to the reuse framework to overcome observed limitations. We believe that our approach to plan reuse can be profitably employed by generative planners in many applied domains. [62]

DISTRIBUTED LEARNING

Most methods of learning in distributed environments are based on gradient descent algorithms that involve changing the weights of the network in order to minimize the difference between the expected and actual input-output behaviors. The successes of such "motion in weight space" methods have been limited due to their inability to capture the implicit constraints of the behavior and properly distribute them among the units of the network. An alternative system has been developed, one based on *motion in constraint space*. It relates the input-output behavior of a connectionist network to a Boolean expression in disjunctive normal form, where each hidden unit of the network learns to detect one of the conjunctive parts of the expression. The potential constraints at a processor are the states of an input configuration that correctly activates the outputs. These constraints are added and removed from the processors in such a way that the correctness of the behavior of the network is maximized. Unlike gradient descent methods, which may become trapped in local minima, or simulated annealing methods, which may need an infinite amount of time to reach a good state, this system determines a correct solution to many problems very quickly. Unlike most traditional "machine learning" algorithms, this system can learn concepts in parallel, is capable of continuously adapting to new information, and is highly resistant to feedback error. Applications to problems such as recognizing (learning) 2-D shapes (such as fish tails) show the potential of the applicability of the method to practical problems. [63, 64]

REFERENCES

[1] John (Yiannis) Aloimonos and Jean-Yves Hervé, "Correspondenceless Detection of Depth and Motion for a Planar Surface." CAR-TR-357, CS-TR-2021, DAAB07-86-K-F073, April 1988.

[2] Radu S. jasinschi, "Intrinsic Constraints in Space-Time Filtering: A New Approach to Representing Uncertainty in Low-Level Vision." CAR-TR-425, CS-TR-2201, DAAB07-86-K-F073, February 1989.

[3] Lingxiao Li and James H. Duncan, "Recovering Three-Dimensional Translational Velocity and Establishing Stereo Correspondence from Binocular Image Flows." CAR-TR-361, CS-TR-2041, DEFG0587ER13782, May 1988.

[4] James H. Duncan and Tsai-Chia Chou, "Temporal Edges: The Detection of Motion and the Computation of Optical Flow." CAR-TR-362, CS-TR-2042, 60NANB7D0722, May 1988.

[5] Minas E. Spetsakis and John (Yiannis) Aloimonos, "Optimal Computing of Structure from Motion Using Point Correspondences in Two Frames." CAR-TR-389, CS-TR-2101, DAAB07-86-K-F073, September 1988.

[6] John (Yiannis) Aloimonos and Dimitris P. Tsakiris, "On the Mathematics of Visual Tracking." CAR-TR-390, CS-TR-2102, DAAB07-86-K-F073, September 1988.

[7] Radu S. Jasinschi, "Towards a Theory of Apparent Visual Motion." CAR-TR-394, CS-TR-2117, DAAB07-86-K-F073, October 1988.

[8] Menashe Brosh, Behrooz Kamgar-Parsi and Behzad Kamgar-Parsi, "The Reliability of the Closed-Form Solution to the Image Flow Equations for 3D Structure and Motion (Quadric Patch)." CAR-TR-397, CS-TR-2123, DAAB07-86-K-F073. October 1988.

[9] Minas Spetsakis and John (Yiannis) Aloimonos, "A Multi-Frame Approach to Visual Motion Perception." CAR-TR-407, CS-TR-2147, DAAB07-86-K-F073, November 1988.

[10] Daniel DeMenthon, "Reconstruction of a Road by Matching Edge Points in the Road Image." CAR-TR-368, CS-TR-2055, DACA76-88-C-0008, June 1988.

[11] Ken-ichi Kanatani and Daniel DeMenthon, "Reconstruction of 3D Road Shape from Images: a Computational Challenge." CAR-TR-413, CS-TR-2167, DACA76-88-C-0008, December 1988.

[12] Kikuo Fujimura and Hanan Samet, "Time-Minimal Paths Among Moving Obstacles." CAR-TR-398, CS-TR-2124, IRI-88-02457, October 1988.

[13] Randal C. Nelson, "Visual Navigation." CAR-TR-380, CS-TR-2087, DAAB07-86-K-F073, August 1988.

[14] David Shulman and John (Yiannis) Aloimonos, "Boundary Preserving Regularization: Theory Part I." CAR-TR-356, CS-TR-2011, DAAB07-86-K-F073, April 1988.

[15] Azriel Rosenfeld, "Explaining Noisy Data: A Qualitative Bayesian Approach." CAR-TR-358, CS-TR-2022, AFOSR-86-0092, April 1988.

[16] Peter Meer, Jean-Michel Jolion and Azriel Rosenfeld, "A Fast Parallel Algorithm for Blind Estimation of Noise Variance." CAR-TR-373, CS-TR-2069, DCR-86-03723, June 1988.

[17] Jean-Michel Jolion and Azriel Rosenfeld, "Cluster Detection in Background Noise." CAR-TR-363, CS-TR-2047, DCR-86-03723, June 1988.

[18] J. John Kim, Dong Yoon Kim and Azriel Rosenfeld, "Mode-Based Cluster Detection." CAR-TR-425, CS-TR-2202, DCR-86-03723, February 1989.

[19] Peter Meer and Isaac Weiss, "Smoothed Differentiation Filters for Images." CAR-TR-424, CS-TR-2194, DCR-86-03723, N00014-88-K-0348, February 1989.

[20] John Canning, "A Note on Mask-Based Least Squares Line Fitting." CAR-TR-384, CS-TR-2095, DAAB07-86-K-F073, August 1988.

[21] Dong Yoon Kim, J. John Kim and Azriel Rosenfeld, "A Robust Method for Fitting a Straight Line to a Noisy Image." CAR-TR-428, CS-TR-2212, DAAB07-86-K-F073, March 1989.

[22] Peter Meer and Yehoshua Y. Zeevi, "The Role of Stimulus Structure in Spatial Hyperacuity." CAR-TR-378, CS-TR-2084, DCR-86-03723, August 1988.

[23] Anne Sutter, Jacob Beck and Norma Graham, "Contrast and Spatial Variables in Texture Segregation: Testing a Simple Spatial-Frequency Channels Model." CAR-TR-381, CS-TR-2091, DCR-86-03723, August 1988.

[24] Azriel Rosenfeld and Jean-Michel Jolion, "Local Operations on Labelled Dot Patterns." CAR-TR-379, CS-TR-2086, DCR-86-03723, August 1988.

[25] Avraham Margalit and Azriel Rosenfeld, "Using Probabilistic Domain Knowledge to Reduce the Expected Computational Cost of Template Matching." CAR-TR-355, CS-TR-2008, DAAB07-86-K-F073, March 1988.

[26] Ramesh Kumar Sitaraman, "The Ordered Matching Problem." CAR-TR-387, CS-TR-2098, DAAB07-86-K-F073, August 1988.

[27] Avraham Margalit and Azriel Rosenfeld, "Using Feature Probabilities to Reduce the Expected Computational Cost of Template Matching." CAR-TR-386, CS-TR-2097, AFOSR-86-0092, August 1988.

[28] Avraham Margalit and Azriel Rosenfeld, "Reducing the Expected Computational Cost of Template Matching Using Run Length Representation." CAR-TR-406, CS-TR-2143, AFOSR-86-0092, November 1988.

[29] Avraham Margalit and Azriel Rosenfeld, "Matching Polygonal Arcs." CAR-TR-418, CS-TR-2174, AFOSR-86-0092, January 1989.

[30] Avraham Margalit and Gary D. Knott, "An Algorithm for Computing the Union, Intersection or Difference of Two Polygons." CAR-TR-350, CS-TR-1995, AFOSR-86-0092, March 1988.

[31] Avraham Margalit and Gary D. Knott, "An Algorithm for Computing the Union, Intersection or Difference of Two Sets of Polygons." CAR-TR-419, CS-TR-2175, AFOSR-86-0092, January 1989.

[32] Behrooz Kamgar-Parsi and Jeffrey L. Jones, "Mapping of the Ocean Floor." CAR-TR-377, CS-TR-2083, N00014-86-K-2034, August 1988.

[33] Lee Spector, James A. Hendler, John Canning and Azriel Rosenfeld, "Symbolic Model/Image Matching in Expert Vision Systems." CAR-TR-370, CS-TR-2060, DAAB07-86-K-F073, July 1988.

[34] Tapio Seppänen, Tapani Westman and Matti Pietikäinen, "Parallel Matching of Attributed Relational Graphs." CAR-TR-376, CS-TR-2073, DACA76-88-C-00008, July 1988.

[35] Tapio Seppänen and Kari Pehkonen, "A Generalized Algorithm for Border Tracking with an Implementation on a Butterfly Parallel Processor." CAR-TR-388, CS-TR-2099, DACA76-88-C-0008, August 1988.

[36] Kari Pehkonen, “The Implementation of Linnainmaa and Harwood’s Pose Determination Algorithm on Shared and Distributed Memory Parallel Computers.” CAR-TR-423, CS-TR-2191, DACA76-88-C-0008, February 1989.

[37] Thor Bestul, “Parallel Quad-Tree Hidden Edge Removal.” CAR-TR-428, CS-TR-2181, DACA76-88-C-0008, January 1989.

[38] E.V. Krishnamurthy and S.G. Ziavras, “Matrix g-Inversion on the Connection Machine.” CAR-TR-399, CS-TR-2125, DACA76-88-C-0008, October 1988.

[39] E.V. Krishnamurthy and S.G. Ziavras, “Complexity of Matrix Partitioning Schemes for g-Inversion on the Connection Machine.” CAR-TR-400, CS-TR-2126, DACA76-88-C-0008, October 1988.

[40] E.V. Krishnamurthy and S.G. Ziavras, “Grid Evaluation-Interpolation on the Connection Maching Using Tensor Products and g-Inversion.” CAR-TR-401, CS-TR-2127, DACA76-88-C-0008, October 1988.

[41] E.V. Krishnamurthy and S.G. Ziavras, “Multivariable Spline-Blending Approximation on the Connection Machine.” CAR-TR-402, CS-TR-2132, DACA76-88-C-0008, October 1988.

[42] Behzad Kamgar-Parsi, J. Anthony Gualtieri, Judith E. Devaney and Behrooz Kamgar-Parsi, “Clustering in Parallel with Neural Networks.” CAR-TR-417, CS-TR-2173, DAAB07-86-K-073, January 1989.

[43] Sotirios G. Ziavras and Larry S. Davis, “Fast Addition on the Fat Pyramid and Its Simulation on the Connection Machine.” CAR-TR-383, CS-TR-2093, DACA76-88-C-0008, August 1988.

[44] C. Allen Sher and Azriel Rosenfeld, “A Pyramid Hough Transform on the Connection Machine.” CAR-TR-421, CS-TR-2182, DACA76-88-C-0008, January 1989.

[45] Peter Meer, “Simulation of Constant Size Multiresolution Representations on Image Pyramids.” CAR-TR-348, CS-TR-1984, DCR-86-03723, January 1988.

[46] Peter Meer, Song-Nian Jiang, Ernest S. Baugher and Azriel Rosenfeld, “Robustness of Image Pyramids under Structural Perturbations.” CAR-TR-354, CS-TR-2007, DCR-86-03723, March 1988.

[47] Jean-Michel Jolion and Azriel Rosenfeld, “Coarse-Fine Bimodality Analysis of Circular Histograms.” CAR-TR-385, CS-TR-2096, DCR-86-03723, August 1988.

[48] Nassir Navab and Azriel Rosenfeld, “Object Detection by Local Averaging and Bimodality Analysis.” CAR-TR-360, CS-TR-2024, DCR-86-03723, May 1988.

[49] Jean-Michel Jolion, Peter Meer and Azriel Rosenfeld, “Border Delineation in Image Pyramids by Concurrent Tree Growing.” CAR-TR-349, CS-TR-1993, DCR-86-03723, February 1988.

[50] Jean-Michel Jolion and Azriel Rosenfeld, “An $O(\log n)$ Pyramid Hough Transform.” CAR-TR-372, CS-TR-2066, DCR-86-03723, July 1988.

[51] Peter Meer, C. Allen Sher and Azriel Rosenfeld, “The Chain Pyramid: Hierarchical Contour Processing.” CAR-TR-376, CS-TR-2072, DCR-86-03723, July 1988.

[52] Azriel Rosenfeld, “An Architecture for Picture Parsing.” CAR-TR-408, CS-TR-2152, DCR-86-03723, December 1988.

[53] Behzad Kamgar-Parsi, Behrooz Kamgar-Parsi and William A. Sander, “Quantization Error in Spatial Sampling: Comparison Between Square and Hexagonal Pixels.” CAR-TR-415, CS-TR-2171, DAAB07-86-K-073, January 1989.

[54] Robert A. Melter and Angela Wu, “Metrically Independent Sets in the Digital Plane.” CAR-TR-352, CS-TR-2002, DCR-86-03723, March 1988.

[55] Azriel Rosenfeld and Angela Y. Wu, “““Digital Geometry” on Graphs.” CAR-TR-371, CS-TR-2065, AFOSR-86-0092, July 1988.

[56] Prabir Bhattacharya and Azriel Rosenfeld, "Contour Codes of Isothetic Polygons." CAR-TR-382, CS-TR-2092, AFOSR-86-0092, August 1988.

[57] Sharat Chandran and David Mount, "Optimal Shared Memory Parallel Algorithms and the Medial Axis Transform." CAR-TR-411, CS-TR-2165, AFOSR-86-0092, December 1988.

[58] S.K. Bhaskar, Azriel Rosenfeld and Angela Wu, "Models for Neighbor Dependency in Planar Dot Patterns." CAR-TR-351, CS-TR-2000, AFOSR-86-0092, March 1988.

[59] Avraham Margalit, "A Parallel Algorithm to Generate a Markov Random Field Image on a SIMD Hypercube Machine." CAR-TR-365, CS-TR-2050, DAAB07-86-K-F073, June 1988.

[60] Ken-ichi Kanatani, "Hypothesizing and Testing Geometric Properties of Image Data." CAR-TR-416, CS-TR-2172, DAAB07-86-K-073, January 1989.

[61] David Harwood, Raju Prasannappa and Larry Davis, "Preliminary Design of a Programmed Picture Logic." CAR-TR-364, CS-TR-2048, DAAB07-86-K-F073, June 1988.

[62] Subbarao Kambhampati, "An Approach to Flexible Reuse of Plans." CAR-TR-367, CS-TR-2054, DAAB07-86-K-F073, June 1988.

[63] John Sullins, "Boolean Learning in Neural Networks." CAR-TR-359, CS-TR-2023, DAAB07-86-K-F073, May 1988.

[64] John Sullins, "Distributed Learning: Motion in Constraint Space." CAR-TR-412, CS-TR-2166, DAAB07-86-K-F073, December 1988.

Note: It was intended to include refs. Nos. [2], [19], [27], and [53], as well as a paper based on Nos. [44] and [50], in full in these Proceedings, but space did not permit their inclusion.

Image Understanding and Robotics Research at Columbia University

John R. Kender[1]
Peter K. Allen
Terrance E. Boult

Department of Computer Science
Columbia University, New York, NY 10027

0 Introduction

Over the past year, the research investigations of the Vision/Robotics Laboratory at Columbia University have reflected the interests of its four faculty members, two staff programmers, and 16 Ph.D. students. Several of the projects involve other faculty members in the department or the university, or researchers at AT&T, IBM, or Philips. We list below a summary of our interests and results together with the principal researchers associated with them. Since it is difficult to separate those aspects of robotic research that are purely visual from those that are vision-like (for example, tactile sensing) or vision-related (for example, integrated vision-robotic systems), we have listed all robotic research that is not purely manipulative.

The majority of our current investigations are deepenings of work reported last year; this was the second year of both our basic Image Understanding contract and our Strategic Computing contract. Therefore, the form of this year's report closely resembles last year's. Although there are a few new initiatives, mainly we report the new results we have obtained in the same five basic research areas. Much of this work is summarized on a video tape that is available on request.

We also note two service contributions this past year. The *Special Issue on Computer Vision* of the *Proceedings of the IEEE*, August, 1988, was co-edited by one of us (John Kender [27]). And, the upcoming IEEE Computer Society Conference on Computer Vision and Pattern Recognition, June, 1989, is co-program chaired by one of us (John Kender [23]).

0.1 Low-Level Vision

0.1.1 Polarization and Specularities

1. New methods for using polarization to segment specular highlights and to separate reflectance components (Larry Wolff [44, 49]).
2. New methods for classifying material surfaces into conductors and dielectrics using the polarization of specular highlights (Larry Wolff, and Terry Boult [45, 50, 51, 52]).

0.1.2 Image Warping

1. A survey of image-warping techniques (George Wolberg [40]).
2. A novel data structure and algorithm for warping to and from arbitrary shapes (George Wolberg [41, 39]).
3. A new, highly efficient, general method for achieving 2-D image warps by separating the transform into two successive 1-D warps (George Wolberg [42, 43]).

0.1.3 Optic Flow, and Rotational Motion

1. New, provably optimal algorithms for determining optic flow based on smoothing splines (Anargyros Papageorgiou, David Lee of AT&T Bell Laboratories, Greg Wasilkowski of the University of Kentucky [30]).

[1] This work was supported in part by the Defense Advanced Research Projects Agency under contracts N00039-84-C-0165 and DACA76-86-C-0024.

2. New algorithms for the smooth interpolation of rotational motions (Ken Roberts, and Kicha Ganapathy and Garry Bishop of AT&T Bell Laboratories [31]).

0.2 Middle-Level Vision

0.2.1 Physical Stereo

1. A unified theory of generalized physical stereo vision for the determination of several first and second order local surface properties (Larry Wolff [46, 53]).
2. A new method for determining local surface orientation from a continuous variation of photometric stereo, called "photometric flow fields" (Larry Wolff [47, 54, 55]).
3. A new invariant within two-camera stereo that allows the determination of the orientation of lines and surfaces in a manner insensitive to baseline measurement error (Larry Wolff [48, 56, 57]).

0.2.2 Regularized Surface Reconstruction

1. A critical study of regularization methodology (Terry Boult [8]).
2. Investigations into the stability and error properties of a new integrated stereo matching, surface reconstruction, and surface segmentation (Terry Boult, Liang-Hua Chen, and Mark Lerner [9, 13]).

0.2.3 Sensory Fusion

1. A new method for classifying textures based on the relative contributions of independent texture methods to a fused texture percept (Mark Moerdler [28]).
2. A method for fusing texture and stereo (Mark Moerdler, and Terry Boult [29]).
3. A working system, now in production, for the spline-based recovery of smooth oceanographic positional information from noisy, conflicting input (Terry Boult, and Barry Allen of Columbia University's Lamont-Doherty Geological Observatory [10]).
4. An initial reexamination of depth-from-focus, for possible use in fusing with stereo and/or texture (Terry Boult).

0.2.4 Shape from Dynamic Shadowing

1. A patent for shape from darkness, a discrete method for deriving surfaces from dynamic shadows (John Kender, and Earl Smith [26]).
2. A parallelizable, optimal algorithm for shape from continuous shadows (Michalis Hatzitheodorou, John Kender [20, 21]).

0.3 Spatial Relations

0.3.1 Representations of Objects

1. An elegant representation for lines in three-space (Ken Roberts [32]).
2. New, robust measures for the error of fit of superquadric models to range data (Ari Gross, and Terry Boult [17, 18]).
3. An investigation into efficiently-computable invariants that quickly relate reflectance information to certain classes of generalized cylinders (Ari Gross).
4. The design and initial implementation of a system to numerically recovery the parametric representations of volumes from multiple types of data, and multiple sensor types (Terry Boult).
5. New algorithms for efficient viewpoint planning (Dino Tarabanis, and Roger Tsai of IBM Watson Laboratory [38]).

0.3.2 Representations of Space

1. A survey of algorithms for the representation of space and free-space path planning (Monnett Hanvey [19]).
2. An analysis of the complexity of efficiently updating of digital distance maps in dynamic, greater than 2-D, environments (Terry Boult [11]).

0.3.3 Theory and Practice of Navigation

1. An analysis of the complexity of topological navigation by landmarks, with applications to the design of sensors and robot instruction languages (John Kender, Avraham Leff, and Il-Pyung Park [24, 25]).
2. Systems issues in real-time robotic navigation (Monnett Hanvey, and Russ Andersson of AT&T Bell Laboratories).

0.4 Parallel Algorithms

0.4.1 SIMD Algorithms

1. Analysis of two novel and several existing algorithms for depth interpolation using optimal numerical analysis techniques (Dong Choi, and John Kender [14, 15, 16]).
2. A method for shape-from-texture based on distortions in image autocorrelation (Lisa Brown, and Haim Schvaytzer of Cornell University [12]).
3. Programming environments and image pyramid emulation for the Connection Machine (Hussein Ibrahim, Lisa Brown, and John Kender [22]).

0.4.2 Pipeline Algorithms

1. Grey-level corner detection in real time (Ajit Singh, Mike Shneier of Philips Laboratories [33, 34]).
2. An integrated system for real-time visual object tracking (Peter Allen [1, 4]).
3. New algorithms for motion perception in real time (Ajit Singh [35, 36, 37]).

0.5 Robotics and Tactile Sensing

0.5.1 Integrated Environments

1. Integrated environments (Peter Allen, Paul Michelman, Ken Roberts, Amy Morishima, and Steve Feiner [2, 5, 6]).

0.5.2 Multi-fingered Object Recognition

1. Haptic recognition via active exploration with a robotic hand (Peter Allen, Ken Roberts [3, 7]).

We now detail these efforts, many of which are documented by full papers in these proceedings. We also include short discussions of work in progress.

1 Low-Level Vision

We have explored three areas of low-level vision, and the results that we have obtained in each of them came via the careful exploitation of new equations, representations, or settings for standard, traditional problems.

1.1 Polarization and Specularities

Prior to this research, the segmentation of specular highlight regions, and the separation of reflected light into diffuse and specular reflection components, could only be solved on dielectric materials (insulators). Additionally, previous algorithms were sensitive to color. However, by using polarization information, specular regions can be identified on both metals and dielectrics on a per-pixel basis, without the use of a segmentation procedure, as long as a controllable polarizing filter can be placed between the camera and the object. Most ordinary light sources are unpolarized, but light reflected off dielectrics tends to be much more polarized and is more easily separated. A few minimal restrictions on the phase angle between light, object, and camera, must apply; these and other qualitative statements have be made quantitatively precise. Initial experimental evidence is very encouraging, on a variety of metals, insulators, and metallic and non-metallic paints and glazes [44, 49].

Closely related to this method--indeed, derived from the same equations--are new methods for classifying material surfaces into conductors or dielectrics by using the polarization of specular highlights. As with the segmentation algorithms, they depend on the empirical determination of the polarization Fresnel ratio. Originally developed for use with point sources, the methods have also been extended to allow their computation to be based more typical extended sources, such as fluorescent tubes [45, 50, 51, 52].

Because both classes of algorithms have the same initial front-end requirements, both can be run in parallel on the same image for simultaneous material classification and separation of reflection components. In addition, a third class of algorithms can exploit further relationships implicit in equations for the polarization of reflected light, in order to determine local surface orientation properties (discussed below, under "middle-level" vision). Thus, the theory unites three very different roles of early vision: object composition, object position and orientation, and environmental lighting and reflections. Under construction is an integrated set of vision algorithms that does these (and other) tasks, called POLARIS, for POLarization And Radiometric Integrated System.

1.2 Image Warping

Many imaging situations call for small local geometric corrections on the retina; many graphic situations call for large ones. Remote sensing, medical imaging, and television commercials with special effects all share the need to elastically deform images to some ground truth or some aesthetic demand. Done in software, image warping can be thought of as a reconfigurable lens system. The existing technology is extensive, but relatively slow, constrained by numerous side conditions, and subject to many errors and aberations. A search for better algorithms resulted first in a comprehensive survey of image-warping techniques [40]).

The literature is largely silent on the problem of efficiently and smoothly mapping between two image regions which are delimited by arbitrary closed curves; such regions do not have the universally assumed four corners. The second result was the specification and verification of an algorithm that instead treats an image region as a collection of interior layers around a skeleton [41, 39]. These layers impose a type of local polar coordinate system which allows each shape to be "unwrapped" into a tree-like representation. Region-to-region warping is then defined by a natural mapping between the two resulting trees. Although there is no a priori way of defining quality of mapping, the results are aesthetically pleasing.

The third and most recent product is a new, highly efficient, general method for achieving 2-D image warps by separating the 2-D transform into two successive 1-D warps [42, 43]. It therefore extends the power of existing hardware systems that perform more limited classes of transformations by similar decompositions. However, this method shows that off-the-shelf hardware, in the form of digital filters with only minor modification for 1-D image resampling, can be used to realize arbitrary mapping functions cheaply and at video rates.

Further work will make use of this approach for performing high-speed elastic matching of deformed images. By using the spatial lookup tables introduced here, improved metrics for the quantification of deformation are possible. Extensions to 3-D may also be straightforward.

1.3 Optic Flow, and Rotational Motion

Optic flow computations are traditionally cast as continuous partial differential equations, but then are solved by discrete difference methods. Although there have been numerous approaches to the problem, differing in both equations and boundary conditions, few results have been obtained concerning the quality of solutions and their error. However, when the problem is cast in the domain of smoothing splines, and if boundary flow values obey Dirichlet constraints, several results are possible [30]. There is a unique solution; sparse, iterative methods can be sued to solve the resulting discrete system; error can be predicted. Further, the Chebyshev method of solution requires little global exchange of information, so it is eminently suited for parallelization. These results appear to be applicable to other low-level vision problems as well.

Looking now instead to the problems of smooth flow by a single object in three space, it is apparent that the understanding and interpolation of rotational motion (as in a "perfect spiral" football pass) is important in computer animation, robot control, and hypothesis-guided computer vision. A new, closed-form algorithm for doing so has been implemented, based on representing motions as quaternions on the unit three-sphere [31]. Resulting displays of interpolated values, and the computer animation sequences based on them, are smoother and more perceptually realistic than existing methods.

2 Middle-level Vision

We have exploited the theory of physical stereo to produce many methods for determining object position, orientation, and curvature. There are at least five now. We have found several powerful alternatives to standard regularization methods, and one of them has led to a non-traditional, one-step method for stereo matching, surface reconstruction, and segmentation. It forms the further basis for a novel stereo-texture fusion system, and holds the promise of further fusion work, possibly with depth-from-focus. The fusion work has already lead to an operational system in use outside the vision community. Research on shape-from-shadows has yielded a patent and optimal, parallelizable, distributed algorithms.

2.1 Physical Stereo

The theory of generalized physical stereo has produced five different applications at the middle levels of vision.

In the first, local surface orientation can be calculated by varying the wavelength and/or the linear polarization of a single incident light source [53]; only two settings of a polarizer are necessary for uniqueness of solution. This is a motionless variation on photometric stereo, and has been one of the earliest results from the theory. However, further study of the method has shown that the process of computing local surface normals can be made to rely simply on the empirical determination of the polarization Fresnel ratio; this is the same parameter necessary for determining the materials of an object and the components of a reflection. Thus, a formerly implicit factor is now seen to be a unifying parameter.

More practically, a second new technique to measure local surface orientation has been based on a more complete theory of the reflection of light. This theory combines the Torrance-Sparrow theory of reflection with the Wolff polarization theory of "quasi-monochromatic" (monochromatically filtered) light [46]. The technique enables surface orientation to be uniquely measured in arbitrary lighting by placing a simple monochrome filter and a linear polarizer in front of the sensor; two images taken at two orientations of the polarizer suffice. The equations that govern the calculations, called the polarization state matrix equations, are elaborate, but they are only a special case of the larger family of generalized physical stereo imaging equations.

These equations can be exploited to derive a third technique: the accurate determination of second order variations of smooth object surfaces as a function of height above the image plane. This technique uses a generalization of surface Hessian methods, which overconstrains the solution for the surface Hessian matrix, giving the second order variations of the smooth surface.

A fourth and very recent method for determining local surface orientation is based on a new imaging concept, the "photometric flow field" across an optic sensing array [47, 54, 55]. Conventional optic flow considers the rate of change in the physical position of the image of an object, as the object actually moves in three-space. In contrast, photometric flow considers the rate of change in the image irradiance of the image of a stationary object, as the illumination geometry moves in three-space instead. Such photometric flow fields can be used to determine local surface orientation and surface curvature. The method may be generalizable to extended light sources.

A fifth corollary to the theory of generalized physical stereo is a method to compute surface orientation from the stereo correspondence of linear features such as polygonal edges, or internal linear markings or texture [48, 56, 57]. It is in contrast to standard stereo, which uses point correspondences to compute the orientation of a plane from the 3-D position of three or more coplanar points. Stereo using line correspondence instead computes the orientation of a plane from the orientations of two or more coplanar lines. In the ideal world these two methods are exactly equivalent. But in the experimental world with measurement error, the errors inherent to measurement of surface orientation from line correspondence stereo does not grow nearly as fast with respect to baseline translation errors or with respect to distance from the baseline. Analysis and Monte Carlo simulations are shown to support this. There may be other vision algorithms which use also profit from the use of equivalent geometric constructions to combat error.

2.2 Regularized Surface Reconstruction

Defining the meaning of "smooth surface" is one of the burdens of surface regularization. In a survey paper, some of the benefits promised by the regularization framework are contrasted to some of its unheralded difficulties, particularly the problems of determining appropriate functional classes, norms, and regularization stabilizing functionals [8]. When regularization is subjectively tested via established procedures of psychology, the results of the methodology applied to the surface reconstruction problem often gives worse results than certain other non-traditional formulations (which are also presented and analyzed).

One of these non-traditional methods provide the basis for a non-heuristic algorithm which simultaneously reconstructs surfaces and segments the underlying data according to the same energy-based smoothness measure [9]. It is founded on the use of reproducing kemel-based splines, which allow efficient calculation of upper and lower bounds on surface energy. The system naturally deals with occluded objects, and also with sharply slanted surfaces, such as roads as seen from a vehicle.

This work on non-heuristic segmentation has been further extended into the development and testing of a new unified approach to stereopsis; it identifies the stereo matching criteria with the already combined non-heuristic reconstruction and segmentation criteria [13]. This energy criterion can be interpreted as a measure of match ambiguity, which is used to rank order all potential stereo matches. Stereo matching, surface reconstruction, and surface segmentation are therefore done in one step, according to one criterion. In tests so far, the method results in fewer unmatchable features than the Marr-Poggio-Grimson method. A parallel implementation is planned, to be followed by comparative performance analyses under various formulations of surface energy, and for various scenes.

2.3 Sensory Fusion

Existing work on the fusion of five different shape-from-texture methods has suggested a novel approach for classifying textures [28]. Each of the methods is tuned to certain image phenomena; the five are shape from spacing, shape from orientation, shape from size, and shape from absolute and relative eccentricity. Given a single texture patch, particularly one under perspective, each method will respond differentially according to the degree it believes the patch possesses cues that the method can exploit to derive shape information. These differential strengths can now all be gathered together as a signature feature vector for the texture. Although such vectors may not have any easily assignable "natural meaning", they can be manipulated in the usual way by standard pattern recognition or image segmentation techniques.

Having found ways of integrating into one process the three steps of stereo perception, and into another process five methods of texture perception, it was inevitable that the two processes themselves would be fused [29]. The resulting system now combines information in two fundamentally different ways, by intra-process and inter-process integration. For standardization reasons, inter-process integration necessarily incorporates a priori assumptions about surfaces, such as degrees and measures of smoothness; it communicates such data in a standardized way via a blackboard organization. In operation, the stereo process uses the relative accuracy and sparseness of the centroid of texels to begin feature localization, later switching to traditional zero-crossings. The work is further characterized by the choice of smoothness measure; roughly it minimizes variation in the 1.5 derivative, not the second. Final integration is achieved by weighting the surface constraints that are output by a process, by an amount that is inversely proportionally to the peak number of constraints a process can output; otherwise stereo, which is denser, would always outrank texture processing.

Applying this fusion technology to a real-world problem led to the successful completion of an operational system for oceanographers. These programs, now in constant use by researchers mapping strctures beneath the ocean floor, integrate navigational and positional information in order to recover the path of smoothly moving ocean vessels. The system's use of smoothing splines is backed by a clever heuristic to ignore faulty outliers in the data. The analysis and review of the project includes documentation of the negative results produced by more standard, "optimal" methods [10]).

Further pursuing the idea of multi-sensor fusion, initial re-implementation and testing has begun on algorithms for depth-from-focus. The experimental project will implement the three leading depth-from-focus algorithms, in order to comparatively determine their cost/accuracy trade-offs. The most efficient one becomes a candidate for further sensor fusion studies.

2.4 Shape from Dynamic Shadowing

The discrete version of a method for extracting surface shape information based from object self-shadowing under moving light sources has been awarded its patent [26].

The continuous version has seen extensive analysis, leading to a optimal, parallelizable algorithm [20, 21]. The two-dimensional problem is solved by decomposing it into a series of one-dimensional slices in the plane of the moving light source; these can be solved in parallel. Each strip is computed using as a basis a family of interpolating splines of an unusual piecewise linear form. The solution is checked against a side system of inequalities in order to preserve the implicit information that points interior to a shadowed region must lie below that shadow line; if the solution fails, a non-linear approximation algorithm accommodates the failing constraints.

The problem has a natural parallelization, not only into slices, but also into hill-and-valley segments; the latter parallelism has been implemented on a loosely coupled network of workstations. A smoothing spline approach has been developed to regularize noisy data. The question of optimal information (i.e., where to put the illuminants) has been solved in some very restricted cases; basically, the problem is dominated by the tangent of the incoming light ray angle. A full analysis of optimal light placement is being pursued.

3 Spatial Relations

We have invented, explored, or improved several representation schemes for objects they occupy and the light they reflect or obscure: lines, polyhedra, superquadrics, generalized cylinders, and sensor models. We have (literally) surveyed the representations of empty space, and how the representations can be efficiently changed as objects move. Even in one dimension, navigation is provably hard; we are examining two and more, in simulation and in the lab.

3.1 Representations of Objects

A new representation for a line in Euclidean three-space has been discovered, which uses only four parameters, the minimal number allowable, and still avoids singularities and special cases [32]. Without sacrificing convenience of computation, it is no longer necessary to represent lines in the more traditional six-parameter forms (such as Plucker coordinates, or point-and-orientation form), although the new representation has the added advantage that it is easy to convert to those forms. The representation, involving two parameters for position and two for orientation, readily generalizes to Euclidean n-space, where it uses 2n-2 parameters: n-1 for position, and n-1 for orientation.

When modeling objects by means of superquadrics, the primary concern in parameter estimation is the proper choice of the error-of-fit measures that control the nonlinear least square minimization techniques. The effectiveness of four such measures was tested on many examples using noisy synthetic data and actual range images, including multiple views of the same object, and including a superellipsoid with negative volume--the latter being an important primitive for constructive solid geometry-based modeling. Existing measures of fit appear inadequate, and a new one that performs significantly bette^r^ was developed and verified [17]. In process is the verification of these predicted differences in complete recovery systems using real data.

A related model of volumes, generalized cylinders, is not nearly as well-defined as superquadrics are. Only certain subclasses appear to be well-specified and well-behaved under reflectance. It would be valuable to be able to quickly and cheaply test an image for the presence of a member of one of these subclasses; these tests could serve as gatekeepers to more expensive algorithms in a general polymorphic shape recovery system. The test need not calculate any parameters; it might exploit invariants that simple confirm or deny membership without any attempt at reconstruction. One such subclass, the straight homogeneous generalized cylinders, can be shown to possess a limited form of such invariants, under various rotational transformations and imaging conditions [18]. The test make good use of contour information as well as image intensity; contour is most useful in recovering the axis, and intensity in recovering any tilt. A prototype system is under construction.

Another new project, the PROVER System (Parametric Representation of Volumes: Experimental Recovery System) is designed to allow numerical recovery of parametric representations from multiple types of data, and multiple sensor types. An important feature of the system is its use of explicit sensor error models. Initial implementations are underway, and a prototype system with restricted parametric representations and data types is already running. The system will be used to develop accurate sensor error models, and will help demonstrate the effect of such models in the recovery of parametric volumes. Because of the significant computation cost of the approach, a parallel implementation is already underway.

Experience with merging multiple sensor data sources usually results in examining the sensor modeling problem from the perspective of the automatic generation of viewpoint, geometric, and sensing constraints. Assuming an assembly or an inspection domain, such an analysis is based on both CAD/CAM object models and low-level sensor models. The emphasis is on the automatic and intelligent handling of partial object descriptions, and partial or total sensor occlusions. The automatic generation of sensor viewpoint is a natural place to begin. The goal is to be able to automatically select a viewpoint for a vision sensor from which features of interest on an object will satisfy particular constraints in the image, among them, visibility. A prototype system has been developed that computes the regions in space where a face of an object occludes the target features [38]. The geometric model of the object is polyhedral, but its faces may be concave and multiply-connected.

3.2 Representations of Space

A survey of some 80 papers dealing with environmental representations of mobile robots has been completed and revised [19]. Most of these representations assume a static two-dimensional world, and a complete bird's-eye knowledge of free space and obstacles. The survey also proposes a taxonomy of this new field: it describes map primitives, such as frames of reference and map symbols, and representations, such as dehydrated free space (mixed polyhedra, and vertex graphs), simple mosaics (tessellations, distance maps, and quadtrees), and reconstituted free space (convex cells, and freeways). There continues to be a relative paucity of results on qualitative, topological navigation, however.

Extending previous work on path planning in dynamic environments using digital distance maps [11], complexity bounds have recently been derived on the constrained distance transform for computing digital distance maps. Further, the method has been extended to handle path planning with spatially varying distance metrics. In particular, digital terrain maps (currently synthetic) can provide auxiliary information (for example, surface height and ground-cover) that affects distance measures in a spatially-varying way. Such spatially varying distance cost problems are relatively frequent, and vertex based algorithms do not generalize well to these problems; their strengths under dynamic updating, however, are being investigated.

3.3 Theory and Practice of Navigation

A model has been formalized for topological navigation in one-dimensional spaces, such as along single roads, corridors, or transportation routes; it demonstrates that the problem is surprisingly difficult computationally [24, 25]. The model includes three levels of abstraction: the concepts and representations of the world itself (a version of "Lineland"), the world as abstracted into symbols and landmarks by an omniscient map-maker, and the world as experienced by a limited navigator who follows the map-makers directions. Having also modeled the navigator's sensors in a primitive way (a sensor here being more like a feature detector), it is straightforward to show that the problem of choosing an effective and efficient subset of sensors for navigation via landmarks is NP-complete. However, simplifying heuristic evaluation functions do exist, and are being explored for their effectiveness. The method has also been extended to a grid-like version of two dimensions, with similar results. It still remains that a "good" set of directions is ill-defined and intractable.

Work on the mobile robot platform of AT&T Bell Laboratories continues; sonar and custom VLSI vertical-edge detecting vision now cooperate, albeit weakly. The edge-tracking Kalman filter has been further refined, and initial models of the corridors and their effect on vertical edge positioning is being investigated.

4 Parallel Algorithms

We have analyzed the performance of the parallelization of several computationally optimal algorithms for depth interpolation; since the problem is typical of others at the low-level of vision, the optimality results should easily transfer. We have invented a particularly simple, accurate, and robust shape-from-texture algorithm based on image autocorrelation that appears to outperform human observation on real scenes of roads, dirt, and grass. We have designed and implemented a near-optimal programming environment for validating parallel pyramid-based SIMD algorithms on the Connection Machine. On our PIPE, we are implementing a system for optic flow determination that fuses the results of intensity correlation methods and spatiotemporal energy methods; the method has already generated a robust grey-level corner detector as an offshoot. The PIPE is fast enough to provide real-time robot arm control information, which we are preparing to demonstrate by the dynamic grasping of moving objects.

4.1 SIMD Algorithms

Many constraint propagation problems in early vision, including depth interpolation, can be cast as solving a large system of linear equations where the resulting matrix is symmetric, positive definite, and sparse. Analysis and simulation of several numerical analytic solutions to these equations for a fine grained SIMD machine with local and global communication networks (e.g., the Connection Machine) shows that two methods are provably optimal in terms of computational complexity [14, 15, 16]). For a variety of synthetic and real range data, the adaptive Chebyshev acceleration method executes faster than the conjugate gradient method, if near-optimal values for the minimum and maximum eigenvalues of the iteration matrix are available.

When these iterative methods are implemented in a pyramidal multigrid (coarse-medium-fine) fashion, using a fixed multilevel coordination strategy, the multigrid adaptive Chebyshev acceleration method executed faster than the multigrid conjugate gradient method again. This appears to be the case because an optimal Chebyshev acceleration method requires local computations only. These methods have now been validate on actual range data.

As a possible front-end to such depth interpolation tasks, a new method for determining local surface orientation was developed from rotationally invariant textures based on the two-dimensional two-point autocorrelation of an image [12]. This method is computationally simple and easily parallelizable, uses information from all parts of the image, assumes only texture isotropy, and requires neither texels nor edges in the texture. Applied to locally planar patches of real textures such as roads, dirt, and grass, the results are highly accurate, even in cases where human perception is so difficult that people must be assisted by the presence of an artificially embedded circular object. However, follow-up extensions attempting to use the method for non-isotropic textures, even with built-in heuristic biases, were not successful. Nevertheless, the algorithm has several exploitable mathematical elegancies, and is amenable to parallel implementation.

As part of our efforts under Strategic Computing, three programming environments that support research on stereo and texture algorithms were developed, in parallel image pyramid style [22]). The current and final programming environment has been designed, installed, and documented; it is a highly efficient pyramid machine emulator that executes those image function primitives on the (University of Syracuse) Connection Machine 2. It cleverly reduces communication contention by an elegant, and probably optimal, embedding of the pyramid within the hypercube network. Mesh operations take only a small fixed amount of overhead proportional to the size of the hypercube; parent/child operations run in a smaller fixed time independent of hypercube size. This code is publicly available.

4.2 Pipeline Algorithms

Real-time "pixel-parallel" versions of a variety of image processing algorithms have now been developed for our PIPE architecture. Based on our past experience with pipelined processors [33], already installed have been algorithms for spatial filtering, spatiotemporal filtering, and pyramid-based spatial processing. Most recently, a novel grey-level template-driven corner detector that combines the advantages of two previously orthogonal approaches has been designed and validated; it executes in real time [34].

One application of these real-time algorithms is in real-time motion tracking [1, 4]. The motion in a scene is found by using spatio-temporal filters on a PIPE. The PIPE is able to update motion energy centroids at 10 HZ and this information is used to update the position of an arm mounted camera which tries to keep the object centered in the field of view. Latencies in the communication system between arm and camera effectively reduce the arm movement rate to 4 HZ. The system is being developed in order to pick moving objects in real-time with our Utah-MIT hand.

Robustness of robotic algorithms is a paramount concern; such reliability can be achieved using an information-fusion based approach. A prototype system is under development that combines multiple cues for a visual measurement, along with an associated confidence; the grey-level corner detector is the first example. Next under investigation is image-flow extraction, using a unified mathematical framework for matching-based and gradient-based techniques [35, 36, 37]). The two techniques are nicely complementary; intensity correlation methods work best in structured scenes, and spatiotemporal energy methods are more suited for textured scenes.

5 Robotics and Tactile Sensing

We have made great progress in integrating a Utah-MIT hand into our robotics testbed. We have developed a number of low-level sensing and actuation primitives that allow one to easily program the hand for simple tasks. In addition, we have been exploring human psychology to understand the ways that humans use active touch and to apply these strategies to our robotics environment.

5.1 Integrated Environments

The Utah/MIT dextrous hand provides a new set of tools to study intelligent touch and grasping. Cartesian-based low level control algorithms for the hand, and a more hybrid scheme using both tendon force and tactile contacts will eventually be part of a comprehensive grasping environment. It will be capable of performing tasks such as locating moving objects and picking them up, manipulating man-made objects such as tools, and recognizing unknown objects through touch. In addition, the integrated programming environment will allow grasping primitives to be included in an overall robotic control and programming system that includes dextrous hands, vision sensors, and multiple degree of freedom manipulators [2, 5, 6].

The system has been used to perform a number of grasping tasks, including pick and place operations, extraction of circuit boards from card cages, pouring of liquids from pitchers, and removing light bulbs from sockets. These tasks have been programmed using DIAL, a parallel, graphical animation language developed by Steven Feiner. DIAL permits task-level scripts which can then be bound to particular sensors, actuators, and methods for accomplishing a generic grasping or manipulation task. We are currently exploring ways to extend an environment such as DIAL to allow programming of a hand to be a first-class primitive in a robotic programming environment.

5.2 Multi-fingered Object Recognition

It requires intelligence and model building to emulate the human ability to recognize objects haptically: that is, by only using external tactile sensors, and internal force and position sensors. However, superquadric models have proven to be surprisingly easy to recover from sparse and noisy sensor data [3, 7]. This appears to be because of their small number of parameters, and consequently their ability o recover the shape descriptions of a very large class of objects. Generic or prototypical recognition strategies are straightforwardly possible.

In experiments, a database of 6 objects consisting of undeformed superquadrics (a block, a large cylinder, a small cylinder) and deformed superquadrics (a light bulb, a funnel, a triangular wedge) was each recovered accurately, with extremely sparse data, typically 30-100 points. This is about 100 times less data than with range sensing, but it has the advantage of not being restricted to a viewpoint that only exposes half the object's surfaces to the sensor. Work is underway to extend this system to include segmented objects, multiple representations of objects, and the dynamic updating of representations.

Using piezo-resistive tactile sensors mounted on the Utah-MIT hand,, we are currently implementing robotic analogs of human haptic shape recovery methods such as shape from enclosure, shape from contour and shape from lateral extent.

6 References

1. Allen, P.K. Real-Time Motion Detection on a Frame Rate Processor. Extended Abstracts of the 41st Annual SPSE Conference, May, 1988.

2. Allen, P.K. "Integrating Vision and Touch for Object Recognition Tasks". *International Journal of Robotics Research 7,* 6 (1988).

3. Allen, P.K. 3-D Modeling for Robotic Tactile Object Recognition. Third International Conference on Robotics and Factories of the Future, August, 1988.

4. Allen, P.K. Real-Time Motion Tracking using Spatio-Temporal Filters. Proceedings of the DARPA Image Understanding Workshop, May, 1989. (These proceedings).

5. Allen, P.K., Michelman, P., and Roberts, K. "An Intelligent Grasping System". *IEEE Computer 22,* 3 (March 1989).

6. Allen, P.K., Michelman, P. and Roberts, K. An Integrated System for Dextrous Manipulation. IEEE International Conference on Robotics and Automation, May, 1989.

7. Allen, P.K., and Roberts, K. Haptic Recognition Using a Dextrous Multi-Fingered Hand. IEEE International Conference on Robotics and Automation, May, 1989.

8. Boult, T.E. Regularization: Problems and Promises. Extended Abstracts of the 41st Annual SPSE Conference, May, 1988.

9. Boult, T.E. and Liang-Hua Chen. Synergistic Smooth Surface Stereo. Proceedings of the International Conference on Computer Vision, December, 1988.

10. Boult, T.E., and Allen, B. Integration of Navigational and Positional Information to Recover the Path of a Smoothly Moving Vessel. Proceedings of the SPIE Conference Sensor Fusion, SPIE, November, 1988.

11. Boult, T.E. "Dynamic Digital Distance Maps". *IEEE Journal of Robotics and Automation* (To appear 1989), .

12. Brown, L.G., and Shvaytser, H. Surface Orientation from Projective Foreshortening of Isotropic Texture Autocorrelation. Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition, June, 1988.

13. Chen, L.-H., and Boult, T.E. Analysis of Two New Stereo Matching Algorithms. Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition, June, 1988.

14. Choi, D.J. *Solving the Depth Interpolation Problem on a Parallel Architecture with Efficient Numerical Methods.* Ph.D. Th., Department of Computer Science, Columbia University, 1988.

15. Choi, D.J., and Kender, J.R. Solving the Depth Interpolation Problem on a Parallel Architecture with a Multigrid Approach. Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition, June, 1988.

16. Choi, D.J., and Kender, J.R. "Solving the Depth Interpolation Problem on a Parallel Architecture with Efficient Numerical Methods". *IEEE Transactions on PAMI* (In revision 1989).

17. Gross, A.D., and Boult, T.E. Error of Fit Measures for Recovering Parametric Solids. Proceedings of the International Conference on Computer Vision, December, 1988.

18. Gross, A.D. Straight Homogeneous Generalized Cylinders: Analysis of Reflectance Properties and a Necessary Condition for Class Membership. Proceedings of the DARPA Image Understanding Workshop, May, 1989. (These proceedings).

19. Hanvey, M. "Environmental Representations for Mobile Robot Navigation". *ACM Computing Surveys* (In revision 1989).

20. Hatzitheodorou, M., and Kender, J.R. An Optimal Algorithm for the Derivation of Shape from Shadows. Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition, June, 1988.

21. Hatzitheodorou, M. The Derivation of 3-*D Surface Shape from Shadows.* Proceedings of the DARPA Image Understanding Workshop, May, 1989. (These proceedings).

22. Ibrahim, H.A.H., Brown, L., and Kender, J.R. Parallel Vision Algorithms--Final Report. Tech. Rept. CUCS-415-89, Department of Computer Science, Columbia University, 1989.

23. Kender, J.R. Some Issues for the Panel on Methodology and Standards in CVPR Research. Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition, June, 1989.

24. Kender, J.R., and Leff, A. "Why Direction-Giving is Hard: The Complexity of Linear Navigation by Landmarks". *IEEE Transactions on Systems, Man, and Cybernetics* (To appear 1989).

25. Kender, J.R., and Leff, A. Why Direction-Giving is Hard: The Complexity of Linear Navigation by Landmarks. Proceedings of the AAAI Spring Symposium Series, March, 1989.

26. Kender, J.R., and Smith, E.M. A Method and an Apparatus for Determining Surface Shape Utilizing Object *Self-Shadowing. Patent Number* 4,792,696. December, 1988.

27. Proceedings of the IEEE. *Special Issue on Computer Vision*, August, 1988. editors Li, H., and Kender, J.R..

28. Moerdler, M.L. Multiple Shape from Texture into Texture Analysis and Surface Segmentation. Proceedings of the International Conference on Computer Vision, December, 1988.

29. Moerdler, M.L., and Boult, T.E. The Integration of Information from Stereo and Multiple Shape-from-Texture. Proc. IEEE Computer Society Conference on Computer Vision and Pattern Recognition, June, 1988.

30. Lee, D., Papageorgiou, A., and Wasilkowski, G. Computational Aspects of Determining Optic Flow. Proceedings of the International Conference on Computer Vision, December, 1988.

31. Roberts, K.S., Ganapathy, S.K., and Bishop, G. Smooth Interpolation of Rotational Motions. Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition, June, 1988.

32. Roberts, K.S. A New Representation for a Line. Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern *Recognition, June, 1988.*

33. Singh, A., and Lala, P.K. "A Multilayer Cellular Architecture for a Highly Parallel VLSI Supercomputer". *IEEE Transactions on Computers* (To appear 1989).

34. Singh, A., and Shneier, M. Grey Level Corner Detection: A Generalization and a Robust Real Time Implementation. Tech. Rept. TR-89-099, Philips Laboratories, Briarcliff Manor, NY, 1989.

35. Singh, A. Information-Fusion: An Approach to Robust Image-Flow. Proceedings of the DARPA Image Understanding Workshop, May, 1989. (These proceedings).

36. Singh, A. Image-Flow Extraction: Information from Discontinuities. Tech. Rept. TR-89-017, Philips Laboratories, Briarcliff Manor, NY, 1989.

37. Singh, A. Aperture Problem: A Review and a New Local Solution. Tech. Rept. TR-89-082, Philips Laboratories, Briarcliff Manor, NY, 1989.

38. Tarabanis, Konstantinos and Tsai, Roger Y. Viewpoint Planning: The Visibility Constraint. Proceedings of the DARPA Image Understanding Workshop, May, 1989. (These proceedings).

39. Wolberg, G. Image Warping Among Arbitrary Planar Shapes. Proceedings of the Computer Graphics International Conference, Geneva, May, 1988. also appears in New Trends in Computer Graphics, editors N. Magnenat-Thalmann and D. Thalmann, Springer-Verlag, pp. 209-218, 1988.

40. Wolberg, G. Geometric Transformation Techniques for Digital Images: A survey. Tech. Rept. CUCS-390-88, Department of Computer Science, Columbia University, 1989. to be published by the IEEE Computer Society Press.

41. Wolberg, G. "A Skeleton-Based Image Warping". *Visual Computer* (1989). to appear.

42. Wolberg, G., and Boult, T.E. Separable Image Warping With Spatial Lookup Tables. Proceedings of the DARPA Image Understanding Workshop, May, 1989. (These proceedings).

43. Wolberg, G. and T. Boult. Separable Image Warping. Computer Graphics (SIGGRAPH 89 Proceeedings), , 1989. (To appear).

44. Wolff, L.B. Segmentation of Specular Highlights From Object Surfaces. Proceedings of the SPIE Optics, Illumination, and Image Sensing for Machine Vision III, 1988, pp. 198-205.

45. Wolff, L.B. Classification of Material Surfaces Using the Polarization of Specular Highlights. Proceedings of the SPIE Optics, Illumination, and Image Sensing for Machine Vision III, 1988, pp. 206-213.

46. Wolff, L.B. Accurate Measurement of Second Order Variations of a Smooth Object Surface. Proceedings of the SPIE Sensor Fusion: Spatial Reasoning/ Scene Interpretation, 1988.

47. Wolff, L.B. Shape From Lambertian Photometric Flow Fields. Proceedings of the SPIE Optics, Illumination, and Image Sensing for Machine Vision III, 1988, pp. 255-262.

48. Wolff, L.B. Measuring the Orientation of Lines and Surfaces Using Translation Invariant Stereo. Proceedings of the SPIE Sensor Fusion: Spatial Reasoning/ Scene Interpretation, 1988.

49. Wolff, L.B. Using Polarization To Separate Reflection Components. Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition, June, 1989. (To appear).

50. Wolff, L.B. "Classification of Material Surfaces using the Polarization of Specular Highlights". *IEEE Transactions on Pattern Analysis and Machine Intelligence PAMI* (In revision 1989).

51. Wolff, L.B. Material Classification and Separation of Reflection Components From Polarization/Radiometric Information. Proceedings of the DARPA Image Understanding Workshop, May, 1989. (These proceedings).

52. Wolff, L.B., and Boult, T.E. Polarization/Radiometric Based Material Classification. Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition, June, 1989. (To appear).

53. Wolff, L.B. "Spectral and Polarization Stereo Methods Using a Single Light Source". *International Journal of Computer Vision* (In revision 1989).

54. Wolff, L.B. Shape Understanding From Lambertian Photometric Flow Fields. Proceedings of the DARPA Image Understanding Workshop, May, 1989. (These proceedings).

55. Wolff, L.B. Shape Understanding From Lambertian Photometric Flow Fields. Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition, June, 1989. (To appear).

56. Wolff, L.B. Accurate Measurement of Orientation From Stereo Using Line Correspondence. Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition, June, 1989. (To appear).

57. Wolff, L.B., and Boult, T.E. Experiments For Determining Surface Orientation From Line Correspondence Stereo. Proceedings of the International Joint Conference on Artificial Intelligence, August, 1989.

Image Understanding at the University of Rochester

Christopher M. Brown
Randal C. Nelson
Computer Science Department
University of Rochester
Rochester, NY 14627

Abstract

The vision group at Rochester is engaged in investigating several aspects of parallel and real-time computer vision with the overall goal of implementing a set of basic sensory-motor behaviors which could serve as a foundation for more sophisticated abilities, and integrating these primary behaviors into multi-modal systems. The emphasis is on behaviors which have relevance to, and can be implemented to work robustly in, a broad range of real-world environments since these are most likely to be useful as fundamental skills.

Our recent work includes commissioning the Rochester Robot, a 3 degree of freedom, two-eyed robot head mounted on a Puma 761 arm, and connected to a Datacube image processor. Several real-time visual behaviors have been implemented, including a vestibulo-ocular reflex (VOR), vergence, and target tracking. Research was also performed in various theoretical aspects of computer vision including parallel evidence combination, parallel object recognition, principal view analysis, and extended Kalman filtering.

1. Reconstruction and Segmentation in Parallel -- Data Fusion

Integrating disparate sources of information has been recognized as one of the keys to the success of general purpose vision systems. In the work of P. Chou and C. Brown [Chou87, Chou88a, Chou88b], data fusion is used to accomplish reliable segmentation and reconstruction in parallel. The computation is formulated as a labeling problem. Local visual observations for each image entity are reported as label likelihoods. They are combined consistently and coherently in hierarchically structured label trees with a new, computationally simple procedure. The pooled label likelihoods are fused with the a priori spatial knowledge encoded as Markov Random Fields (MRFs). The a posteriori distribution of the labelings are thus derived in a Bayesian formalism. A new inference method, called Highest Confidence First (HCF) estimation, is used to infer a unique labeling from the a posteriori distribution. HCF has the computational advantages of efficiency and predictable running time. It degrades gracefully, and follows a least-commitment strategy. ITs results are consistent with observable evidence and a priori knowledge, and (not least) it out-performs other known methods. The comparative performance of HCF and other methods has been empirically tested on synthetic and real scenes, using both intensity and sparse depth data for sensor fusion experiments.

2. Principal Views

During 1988, Nancy Watts pursued research leading to progress in the difficult problem of characterizing the different views presented to an observer (in either perspective or orthographic projection) by a non- convex polyhedron. This work was a continuation of her earlier work which produced an algorithm for computing all views of a convex polyhedron. This research is still in progress, but has resulted in a paper presented at ICPR [Watt88].

The usual approach to this problem is from the point of view of abstract computational geometry, in which existence proofs and non- constructive techniques based on them abound. Watts' work is distinguished by her desire to specify data structures and algorithms that will not only enumerate views but will allow them to be used in applications. Her earlier work interfaced nicely with a graphics program that produced sample images from any given view region for convex polyhedra.

To stand a chance of success in the violently combinatorial and geometrically complex situation that arises with non-convex objects, Watts restricted her work to a large class of objects that includes many every-day manufactured objects. She was able to catalog the incidence phenomena that take place in the projective process, and use this information to design data structures and algorithms for characterizing the aspect graph of objects in her class. The main computational tool is "plane sweeping", which is a way to keep track of regions of 3-space as their vertices are encountered by a plane sweeping through space.

3. Parallel Object Recognition

Paul Cooper worked on the general problem of parallel object recognition. The particular instance chosen for implementation was the recognition of Tinker Toy objects from images.

One development was a solution to the Tinker Toy matching problem that accommodates the geometric parameters of the object. That is, an object is recognized not just from its topology, but also from the geometric characteristics such as the lengths of the pieces and the angles between the pieces at the junction. The key to this solution was framing the labeling problem so that the geometry of the junctions was implicitly encoded. When framed in this manner, the labeling problem can be solved by the application of the massively parallel constraint satisfaction network developed earlier by Swain and Cooper [Swai88]. The application of the network to the Tinker Toy matching problem with geometry is reported in [Coop88b].

Another development was the use of domain specific information to generate optimized constraint satisfaction nets. Implementing the general form of Swain and Cooper's [Swai88] network to solve Tinker Toy matching proved infeasible due to resource requirements. But a way of exploiting the characteristics of the Tinker Toy matching domain in order to optimize the general network was developed, resulting in a smaller network that could be (and was) built. Later, a way of specifying domain characteristics for arbitrary domains, was discovered, allowing optimized networks to be built, in an analogous manner, for any domain. This work is described in two papers by Cooper and Swain [Coop88b, Coop88c].

The final and most important development was a method for matching Tinker Toys that could incorporate inexact and uncertain information. The crux of this solution was the use of coupled Markov Random Fields to solve, simultaneously, the segmentation and matching problems in the Tinker Toy domain. The architecture of the solution is essentially the same as that of previous work [Coop88a, Coop88b], in that the problem is framed as a labeling problem in the unit/value connectionist design style. However, instead of adopting discrete constraint satisfaction [Coop88b] or discrete connectionist relaxation [Coop88a] as the formal machinery, Markov Random Fields (MRFs) are used. With a MRF representation, priors are combined with hypothesis likelihoods to yield a probability distribution of solutions with rigorous Bayesian semantics. The result is a scheme that can recognize both occluded objects, and ones obscured by noisy data.

A final report on the MRF project as well as everything else will be available in the thesis [Coop89].

4. The Rochester Robot and Gaze Control

During the summer of 1988 a team at the University of Rochester commissioned the Rochester robot, consisting of a Unimate PUMA 761 arm and a three-dof, two-eyed robot head [Brow88a, Brow88b, Ball88, Olso88]. The robot is interfaced with a Datacube MaxVideo image processor which allows implementation of real-time visually controlled behaviors. A number of basic reflexes were implemented for controlling the gaze of the robot, including adaptive tracking, vergence, and a vestibulo-optic reflex. We believe that an active vision system can use such skills to advantage as building blocks for behavior. We also maintain that appropriate active control of the visual system can significantly simplify visual processing in many cases. An example of this is the kinetic depth mechanism discussed below.

The first gaze control mechanism developed was the vestibulo-ocular reflex or VOR. This is a reflex that stabilizes images on the retina to compensate for head motion. Stabilization aids low-level vision by keeping edges sharp, and reducing motion blur. We noticed that motion blur could contribute positively to image segmentation if it could be used to blur objects that were NOT to be attended. Thus it would reduce high-frequency image phenomena such as edges and textures that are distracting to segmentation algorithms.

Rimey and Brown, on the suggestion of Ballard, implemented the functional equivalent of the VOR using a builtin facility of the robot command language, and implemented a motion-blur amplifier in MaxVideo. The results are gratifying -- the moving head causes severe blur of scene components that are not fixated, thus throwing the fixated objects into strong relief.

A second development was a system due to Tilley and Ballard that successfully tracks moving objects. This real-time adaptive tracking mechanism effectively implements a "smooth pursuit" system. The basic idea is to extract an image patch and use it, with some pre-processing, as a real-time correlation template. This worked fairly well, allowing the robot to "lock onto" a point of interest and maintain a stable gaze while the object or the robot moved. As in the case of the VOR, such stabilization of the region of interest can both simplify analysis of the objects at that point, and aid segmentation through motion blurring of irrelevant background details.

A third reflex was a gross vergence system implemented by Olson. Here, vergence is based on a global disparity calculated between subsampled left and right images. Thus it reflects large-scale image phenomena, not high-resolution ones. The work is reported in [Olso88]. The basic image-processing mechanism for implementing the global disparity calculation is the cepstral filter, which is defined as the fourier transform of the logarithm of the power spectrum. This operation is equivalent to correlating the left and right images, using a nonlinear operation to sharpen the correlation peaks. The computation leads to a measure of global disparity in image x and y, which is translated into radians of rotation via a small-angle approximation. Applying the compensating rotation verges the camera.

An example of using gaze control to simplify visual processing is the kinetic depth mechanism [Ball88]. The object of this work is to produce a depth map in real time using optic flow produced by head motions and knowledge about those head motions. The idea is simply that the retinal flow of a patch of image of a static 3-D scene induced by a head motion depends on the depth of the scene producing the image patch and upon the head motion. It also varies with the fixation of the eyes. If the eyes fixate a patch of scene during head motion (using either tracking or vestibular feedback for example), optic flow is zero at that point. Thus with fixation kinetic depth provides depth information relative to the fixation point. A real-time kinetic depth algorithm was successfully implemented using a simple Horn-Schunck optic flow calculation, and table lookup in our MaxVideo hardware.

Gaze control in biological organisms involves several processes that may interact in non-trivial ways. Brown has developed a model and a simulator for studying the interaction of five basic gaze control processes: saccadic motion, smooth pursuit, vergence, vestibulo-ocular reflex, and head motion. This work has allowed us to formulate and test control strategies for integrating these interrelated behaviors into a unified system.

5. Kalman Filtering and Optimal Estimation Experiments

Recent work by Brown addresses the application of Extended Kalman Filtering (EKF) to basic visual behaviors, and in particular, to the problem of tracking an object moving in a complex manner. Kalman filtering is a form of optimal estimation characterized by recursive (i.e. incremental) evaluation, an internal model of the dynamics of the system being estimated, and a dynamic weighting of incoming evidence with ongoing expectation that produces estimates of the state of the observed system. The primary reference is [Bar88].

The basic Kalman filter is an iterative loop. Its input is the system measurements; its a priori information is the system dynamics and noise properties of system and measurement; and its useful outputs are the *innovation* (the difference between the predicted and observed measurement, by which the filter's performance may be quantified), and the estimated system state updated. The (first order) Extended Kalman Filter (EKF) is a version of the Kalman filter that deals with nonlinear dynamics or nonlinear measurement equations, or both. It linearizes the problem around the predicted state (a second-order EKF makes a second-order approximation). The basic filter control loop still applies, but measurements are predicted using a nonlinear measurement equation h, and in the calculations for filter gain, state update, and covariance update, the Jacobean of h is used. Likewise state prediction is accomplished using the nonlinear state equation f and the state prediction covariance is computed using the Jacobean of f. These generalizations call for extensions to the EKF data structure in which functions (as opposed to matrices) are attached to the filter.

Brown has applied the EKF to the problem of tracking a moving target from a moving observer when the target may *maneuver*, i.e. depart from the basic, steady-state, "normal" dynamic behavior. A method termed *variable dimensional filtering* was used, which essentially substitutes a different, higher order filter when departure from the modelled trajectory is detected. He also addressed ways of tracking a moving target against a cluttered background, comparing the performance of *track splitting*, *nearest neighbor standard filter*, and *probabilistic data association filter* approaches under various conditions. The results of this work are reported in a paper in these procedings.

6. Visual Navigation

In fall 1988 Randal Nelson joined the faculty and the vision group at Rochester having completed his PhD at the University of Maryland. The dissertation research involved the description and implementation of a set of foundational abilities for visual navigation. In particular, visual methods were described for performing passive navigation, obstacle avoidance, and homing in general, real-world environments [Nels88d]. This work fits nicely within the framework of active vision which the group is currently pursuing and, since he intends to continue work along similar lines, a summary of the dissertation results follows.

Passive navigation is a process by which a system obtains information about its rotation and translational motion. This information is useful in navigation to stabilize and direct the motion of the system. Visual methods attempt to obtain the motion parameters from a time-series of images [Gibs50, Praz80, Horn81, Hild83, Lawt83, Long84, Koen86]. The problem is hard because solution methods tend to be extremely sensitive to small errors in the input, while accurate image flow or point correspondence information is difficult to obtain [Tsai81, Adiv85, Anan85, Nage86, Verr87]. The dissertation shows how accurate motion parameters can be obtained from inaccurate flow data by utilizing image information over the entire visual sphere [Nels88a]. Essentially global topological constraints are used to stabilize the process. It is interesting to note that such spherical images are available to flying insects such as bees and dragonflies, so there is a biological precedent.

Obstacle avoidance refers to the ability of a system to move about in the environment without striking the objects in it. This is a fundamental navigational behavior. It is shown that computation of divergence-like properties of the visual flow field provides qualitative cues which are invariant under rotation of the system and which are sufficient to permit the system to avoid collisions. The method is applicable in general environments, the only requirement being the presence of sufficient visual texture to allow the image flow to be roughly approximated. Empirical measurements show that sufficient texture is present in ordinary objects such as stones, trees, and faces. The method was implemented and used successfully to control the motion of a camera in various environments.

Homing is the process by which an autonomous system guides itself to a particular location on the basis of sensory input. This is a slightly more sophisticated, but still fundamental navigational ability. In the dissertation, a method of visual homing using an associative memory [Hint81, Ackl85, Rume86, Smol86,] based on a simple pattern classifier is described [Nels88c]. Homing is accomplished without the use of an explicit world model by utilizing direct associations between learned visual patterns and system motor commands. The technique is analyzed in terms of a pattern space and conditions obtained which allow the system performance to be predicted on the basis of statistical measurements on the environment. The method was implemented and used to guide a robot-mounted camera in a three-dimensional environment. This work is described in the paper *visual Homing Using an Associative Memory* in these proceedings.

7. References

[Ackl85] - D. H. Ackley, G. E. Hinton, and T. J. Sejnowski, A learning algorithm for Boltzmann machines, *Cognitive Science* 9, 1985, 147-169.

[Adiv85] - G. Adiv, Inherent ambiguities in recovering 3-D motion and structure from a noisy flow field, Proc. 1985 IEEE Computer Society Conference on Computer Vision and Pattern Recognition, 70-77.

[Anan85] P. Anandan and R. Weiss, Introducing a smoothness constraint in a matching approach for the computation of optical flow fields, Proc. Third Workshop on Computer Vision: Representation and Control, 1985, 186-194.

[Ball88] D. H. Ballard and A. Ozcandarli, Eye fixation and early vision: Kinetic depth, Proc., 2nd IEEE Int'l. Conf. on Computer Vision, December 1988.

[Bar88] Y. Bar-Shalom and T. E. Fortman, *Tracking and Data Association*, Academic Press, 1988.

[Bert85] *Adaptive Mechanisms in Gaze Control*, Edited by Berthoz and Melvill-Jones, Elsvier 1985,

[Broo86] R. A. Brooks, Achieving artificial intelligence through building robots. A. I. Memo 899, Massachusetts Institute of Technology Artificial Intelligence Laboratory, May, 1986.

[Brow88a] C. M. Brown (Ed), with D.H. Ballard, T.G. Becker, R.F. Gans, N.G. Martin, T.J. Olson, R.D. Potter, R.D. Rimey, D.G. Tilley, and S.D. Whitehead, The Rochester robot, TR 257, Computer Science Dept., U. Rochester, August 1988.

[Brow88b] C. M. Brown and R.D. Rimey, Coordinates, conversions, and kinematics for the Rochester Robotics Lab, TR 259, Computer Science Dept., U. Rochester, August 1988.

[Coop88a] Paul R. Cooper, Structure Recognition by Connectionist Relaxation: Formal Analysis, Proceedings Canadian Artificial Intelligence Conference CSCSI-88, Edmonton, Alberta, June 1988.

[Chou87] P. B. Chou, and C.M. Brown, Probabilistic information fusion for multi-modal image segmentation, Proc., Int'l. Joint Conf. on Artificial Intelligence, Milan, August 1987.

[Chou88a] P. B. Chou and C.M. Brown, Multimodal reconstruction and segmentation with Markov Random Fields and HCF optimization, Proc., DARPA Image Understanding Workshop, Boston, MA, April 1988; submitted (August 1988), *Int'l. J. of Computer Vision.*

[Chou88b] P. B. Chou, *The theory and practice of Bayesian image labeling,* TR 258 and PhD. Thesis, Computer Science Dept., U. Rochester, August 1988; submitted, *Int'l. J. Computer Vision.*

[Coop88b] Paul R. Cooper and Michael A. Swain, Parallelism and Domain Dependence in Constraint Satisfaction, University of Rochester Technical Report TR 255, December 1988.

[Coop88c] Paul R. Cooper and Michael A. Swain, Domain Dependence in Constraint Satisfaction, submitted for publication, December 1988b.

[Coop89] Paul R. Cooper, *Parallel Object Recognition for Structure,* Ph.D. Thesis, May 1989.

Gibs[50] J. J. Gibson, *The Perception of the Visual World,* Houghton Mifflin, Boston, Mass., 1950.

[Hild83] - E. C. Hildreth, *The Measurement of Visual Motion,* The MIT Press, Cambridge, Mass., 1983.

[Hint81] - G. E. Hinton and J. A. Anderson (editors), *Parallel Models of Associative Memory* Lawrence Earlbaum Associates Inc., Hillsdale, N.J., 1981.

[Horn81] - B.K.P. Horn and B.G. Schunck, Determining optical flow, *Artificial Intelligence* **17**, 1981, 185-204.

[Koen86] J. J. Koenderink, Optic flow, *Vision Research* **26**, 1986, 161-180.

[Lawt83] - D.T. Lawton, Processing translational motion sequences, *CVGIP* **22**, 1983, 116-144.

[Long84] H. C. Longuet-Higgins K. and Prazdny, The interpretation of a moving retinal image, *Proceedings of the Royal Society of London* **B 208**, 1984, 385-397.

[Marr82] D. Marr, *Vision,* W. H. Freeman & Company, San Francisco, 1982

[Nage86] H. H. Nagel and W. Enkelmann, An investigation of smoothness constraints for the estimation of displacement vector fields from image sequences. *IEEE Trans. PAMI* **85**, Sept. 1986, 565-593.

[Nels88a] R.C. Nelson and J. Aloimonos, Finding motion parameters from spherical flow fields (or the advantages of having eyes in the back of your head) *Biological Cybernetics* **58**, 1988, 261-273.

[Nels88b] R.C. Nelson and J. Aloimonos, Using flow field divergence for obstacle avoidance in visual navigation, *IEEE transactions on PAMI,* To appear.

[Nels88c] R.C. Nelson Visual homing using an associative memory, *International Journal of Computer Vision,* Under review.

[Nels88d] R.C. Nelson *Visual Navigation*, PhD dissertation, University of Maryland, 1988.

[Olso88] T. J. Olson, and R.D. Potter, Real-time vergence control, TR 264, Computer Science Dept., U. Rochester, November 1988; submitted, IEEE Conf. on Computer Vision and Pattern Recognition.

[Pete88] *Head Control*, Ed. by E. Peterson, Oxford U. Press 1988.

[Poes88] Massimo Poesio, Using Markov Random Fields for Segmentation in the Tinker Toys Domain, University of Rochester Internal Report, December 1988.

[Praz80] K. Prazdny, Egomotion and relative depth map from optical flow, *Biological Cybernetics* **36**, 1980, 87-102.

[Rume86] D. E. Rumelhart, G. E. Hinton, and R. J. Williams, Learning internal representations by error propagation, in *Parallel Distributed Processing*, D. E Rumelhart and J. L McClelland, editors, The MIT Press, Cambridge Mass., 1986, 318-362.

[Smol86] - P. Smolensky, Information processing in dynamical systems: foundations of harmony theory, in *Parallel Distributed Processing*, D. E Rumelhart and J. L McClelland, editors, The MIT Press, Cambridge Mass., 1986, 195-281.

[Swai88] Michael A. Swain and Paul R. Cooper, Parallel Hardware for Constraint Satisfaction, Proceedings AAAI-88, American Association of Artificial Intelligence, St. Paul, Minn., August 1988.

[Tsai81] R. Y. Tsai and T. S. Huang, Estimating 3-D motion parameters of a rigid planar patch I, *IEEE ASSP* 30, 1981, 525-534.

[Verr87] A. Verri and T. Poggio, Against quantitative optical flow, International Conference on Computer Vision, June 1987, 171-180.

[Watt88] N. Watts, Calculating the principal views of a polyhedron, TR 234, Computer Science Dept., U. Rochester, December 1987; Proc., Int'l. Conf. on Pattern Recognition, November 1988.

KNOWLEDGE BASED VISION FOR TERRESTRIAL ROBOTS

Daryl T. Lawton
Advanced Decision Systems
1500 Plymouth Street
Mountain View, CA 94043

Tod S. Levitt
Advanced Decision Systems
1500 Plymouth Street
Mountain View, CA 94043

1. INTRODUCTION

The Knowledge Based Vision Project [Lawton et.al. - 86, Lawton et.al. - 87, Lawton et.al. - 88] is concerned with developing terrain recognition and modeling capabilities for autonomous land vehicles. One of the basic functions of the vehicle is to elaborate this terrain map of the environment. Another is to successfully navigate through the environment using landmarks. For functioning in realistic outdoor environments, we assume vehicles with laser range finders, controllable cameras, and limited inertial sensing. The range finder is used for mapping and navigating through the immediate environment. The cameras are used for object recognition and recognizing distant landmarks beyond the access of the range sensor. We also assume realistically limited perceptual and object recognition capabilities. In particular, it will see things that it won't be familiar with and can't recognize, but which can be described as stable visual perceptions. The vehicle will not always be able to recognize the same object as being identical from very different points of view. It will have limited, inexact, and undetailed a prior terrain information generally in the form of labeled grid data. Critical questions for this work are how to:

- organize memory about local coordinate systems of landmarks as the primary means of defining locations
- account for the nature of visual events, and, in particular, use representations that allow for very poor range and angular measurements, while making full use of the extractable strong visual cues (e.g. occlusion) as primary data in visual memory representations
- maintain memory structures that associate local landmark systems along paths of motion that the robot executed when it saw the landmarks
- admit inference processes over visual memory that robustly perform navigation and guidance despite the poor quality of the quantitative data.

Two particular types of terrain recognition and processing have been explored. The first involves creation of a predicted scene from a prior terrain information. This is then used to direct grouping processes which find predicted structures such as road regions, horizon lines, a terrain patch discontinuities [Lawton et.al. - 87]. The second type of processing involves developing an qualitative environmental map from a freely moving robot without necessarily using any a prior terrain information [Levitt - to appear]. This uses a generic terrestrial scene model which includes several constraints on the formation of perceptual groups based upon the relative direction of gravity, the horizon line determined by the orientation to the immediate ground plane, and the projected egocentric directions from the observer on this plane. The underlying spatial representation and planning mechanisms, referred to as Qualitative Navigation, relates locally obtained viewer-centered representations of the environment into a perceptually-based map that can be used for navigation.

2. PERCEPTUAL ORGANIZATION

Perceptual organization or grouping concerns how local image structures are combined into globally coherent ones, generally using criteria such as continuity and completion. Grouping is of critical importance for knowledge based vision since the predictions generated by object models tend not to appear automatically in images. Extracted groups can also be used to index into a large database of a prior models. For our work, grouping has been used to match the predictions from a prior map data, which can be very coarse and to extract stable perceptual structures which serve as landmarks to incorporate into a map of the world formed by a freely roaming vehicle. We have developed three different types of groupers: the measure based grouper, the hierarchical grouper, and the Hopfield grouper.

The measure based grouper is a generalization of the edge linker developed by Martelli and others [Martelli - 72,76]. This treats grouping as a search based relational query over a data-base of spatial objects such as extracted edges and regions. A group is specified by an initial set of seed objects; global constraints on shape and object attributes which can be incorporated into the group; local constraints on neighborhoods which specify successor objects; and an optimization function which is based upon the type of group.

The hierarchical grouper [Lawton and McConnell - 87] was developed to address limitations of the measure based grouper. The first was how to extract groups from an image without initially specifying the type of group or an explicit optimization criteria. To do this group-type specific rules were developed for extracting an initial set of seed objects. The other was how to group objects which were related but separated by large distances in an image. To achieve this, the database of extracted image objects was partitioned into a pyramid of overlapping areas. Groups are then combined at different levels of the pyramid corresponding to increasing amounts of spatial separation in an image.

In the Hopfield grouper [Gelband and Lawton - 88] the consistencies and constraints which define perceptual grouping criteria are encoded in the coefficients of an energy functional whose state variables are the perceptual links between image tokens. These links may be either on or off, that is, existent or nonexistent. Minimization of the energy functional results in the formation of perceptual groups of image tokens which are optimal with respect to the grouping constraints. This work is modeled after Hopfield's neural network approach [Hopfield - 84, Hopfield and Tank - 85] to solving non-polynomial optimization problems. A preprocessing stage is used to first extract symbolic tokens from an image or time-sequence of images, and then to reduce these tokens to primitive tokens. A local or global energy minimization procedure is then used to form stable perceptual groups, i.e., linked clusters of tokens. A control process freezes stable groups by forming permanent links from the variable links; it then allows the minimization grouping process to continue at a higher level to produce larger coherent structures. This recursive combination of perceptual groups corresponds to a hierarchy of processing performed by the network.

4. QUALITATIVE NAVIGATION

Qualitative Navigation [Levitt - to appear, Kuipers and Levitt - 88, Levitt and Lawton - 89, Levitt et.al. 87] is a multi-level theory of spatial representation of the environment based upon the observation and re-acquisition of distinctive visual events, i.e., landmarks. The representation provides the theoretical foundations for a visual memory database that includes coordinate free, topological representation of relative spatial location, yet smoothly integrates available metric knowledge of relative or absolute angles and distances. We have developed qualitative path planning and execution algorithms that are robust in the face of very coarse or absent measurements of range to, and angle between, landmarks. Rules and algorithms are presented that, under the assumption of correct association of landmarks on re-acquisition (although not assuming landmarks are necessarily re-acquired) provide a robot with navigation and guidance capability. The ability to deduce or update a map of the environment, a posteriori, is a by-product of the inference process. In order to demonstrate our claims, we have built a qualitative navigation simulator, QUALNAV, that provides a software laboratory for experimenting with spatial relationships in visual memory, simulated visual acquisition of landmarks, and their relationship to path planning and execution.

A key contribution of this work is the realization of true local coordinate systems, and a theory that makes them computationally useful to a mobile robot. Using local coordinate systems, which we call viewframes, a robot can navigate about its environment, determining its relative location in the world with essentially a constant error. In particular, there is no multiplicative accumulation of error in location that is the shortfall of all schemes that depend upon a global coordinate system for location.

To accomplish this, the notion of a geographic "place" is defined in terms of data about visible landmarks. A place, as a point on the surface of the ground, is defined by the landmarks and spatial relationships between landmarks that can be observed from a fixed location. More generally we can define a place as a region in space, in which a fixed set of landmarks can be observed from anywhere in the region, and relationships between them do not change in some appropriate qualitative sense. Data about places is stored in structures called viewframes, boundaries and orientation regions.

Viewframes provide a definition of place in terms of relative angles and angular error between landmarks, and very coarse estimates of the absolute range of the landmarks from our point of observation. Boundaries and orientation regions provide a more qualitative definition of place. Both concepts allow us to localize ourselves in space relative to a set of observed landmarks, without necessarily using a priori map data.

Viewframes allow us to localize our position in space relative to observable local landmark coordinate systems. In performing a viewframe localization, we can make use of observed or inferred data about our approximate range to landmarks. Errors in ranging and relative angular separation between landmarks are smoothly accounted for. A priori map data can also be incorporated.

If we drop all range information, we can still use the notion of boundaries to determine our qualitative position relative to other landmarks. A pair of landmarks creates a virtual division of the ground surface by the line connecting the two landmarks. The observable relative orientation of the landmarks, i.e., the left-to-right order of the pair of landmarks, indicates which side of the landmark-pair-boundary (LPB) we are on. A set of LPBs with orientations determines a region on the ground called an orientation region. LPBs can be derived by considering pairs of landmarks from viewframes; in this case we speak of the set of orientation regions induced by or associated to the viewframe.

Viewframes are extracted by an ongoing, multi-stage process. A panoramic set of images is obtained since the localization is aided by multiple landmarks distributed in multiple directions relative to the robot. The hierarchical grouper is used to find a set of restricted perceptual groups including long lines aligned with gravity, junctions, repeated patterns of parallel lines. These correspond to potential landmarks. We assume that the robot translates from viewframe to viewframe with significant changes in robot direction being a useful criteria for extracting a new viewframe. The translational motion assumption isn't necessary but it does simplify matching. The extracted groups are then tracked along the determined translational flowlines. The number of frames over which a landmark is tracked is associated with a landmark to establish it's reliability and strength. The disappearance of landmarks due to occlusion is used to establish when a new viewframe should be extracted.

5. IMAGE UNDERSTANDING ENVIRONMENTS

We have done research in developing software environments for image understanding research and applications [Lawton and McConnell - 88]. This work was initially motivated by supporting internal developments and for expediting technical transfer as a part of the autonomous land vehicle effort. In the last year it has become an area of research and development in it's own right of significance to DARPA. IU environments support shared development across multiple researchers and projects. They make possible the rapid prototyping of applications. The programming constructs used in such environments may help in developing constructs for machine independent development. The also may supply a common set of tools and standards across the IU community.

We initially developed the PowerVision environment [McConnell et.al. - 88] on the SYMBOLICS LISP machines using FLAVORS. It was characterized by a small number of modular components and programming constructs built around objects commonly used in image understanding such as images, curves, regions, junctions, and groups. It consisted of a macro language called DEFIU for writing code and manipulating defined objects in terms of local neighborhood-level operations. It utilized various types of system databases a database query language, a display language, display windows, and several types of browsers to interact with objects. These components were highly independent and could be combined in creative ways by programmers.

View [Edelson et.al. - 88] and Shark [Dye et.al. - 88] were developed to produce a machine independent image understanding environment. This was motivated in part by the transfer of work from LISP-based processors such as SYMBOLICS to less expensive and more general computer workstations such as SUNS. SHARK is a commonlisp/CLOS based [Bobrow et.al. - 87] toolkit for building user interfaces. It is currently built on top of SUN/NEWS but will be ported to X-windows. Shark supports general display objects such as image-display windows, browsers, menus, tables, and graphs. VIEW is a CLOS based set of image understanding constructs which are machine independent. VIEW defines a general inheritance hierarchy of image understanding objects which consist, at the highest level, of two general types of objects: spatial objects and database objects. It is possible to associate methods with these general objects and have method inheritance and combination occur for specialized instances. Thus, convolution, when defined for general spatial objects specializes to deal with arbitrarily shaped, registered, multi-dimensional objects. The uniformity provided by CLOS allows for access to other CLOS tools developed at ADS for Bayesian inference nets and geometric modeling.

A recent effort is porting the CommonLisp Based IU environment onto the Apple MAC II series of personal computer workstations. This is motivated by several factors. The MAC II is an open architecture which is extendable by inexpensive co-processors and, increasingly, low cost video cameras, digitizers and processing boards as desktop video and image processing becomes a commercial reality. The MAC II has an extensive, reliable, and highly optimized user interface with an enormous number of commercially developed software products. We have found that the MAC II was superior in terms of constructing an interactive user interface, for linking video co-processors and memory devices into a coordinated environment. Major difficulties were found to arise from the lack of an operating system supporting virtual memory, multi-tasking, and interprocess communication. Image based computations in the current version of Coral CommonLisp were also much too slow to support a researcher working on problems of any realistic scale. Nonetheless, these factors are mostly premature and not intrinsic to the MAC: MACH and AU/X are now supported and Apple has bought Coral CommonLisp. Important lessons from our work in IU Environments is the importance of machine independence in developing an environment. It is also important to utilize, when ever possible, commercially (or community) supported interfaces and programming environment tools.

ACKNOWLEDGMENTS

This document was prepared by Advanced Decision Systems (ADS) of Mountain View, California, under U.S. Government contract number DACA76-85-C-0005 for the U.S. Army Engineer Topographic Laboratories (ETL), Fort Belvoir, Virginia, and the Defense Advanced Research Projects Agency (DARPA), Arlington, Virginia. The authors wish to thank Sachi Toyofuku for providing administration, coordination, and document preparation support.

References

[Bobrow et.al. - 87] D. G. Bobrow, L. G. DeMichiel, R. P. Gabriel, S. Keene, G. Kiczales, and D. A. Moon, "Common Lisp Object System Specification", 1987.

[Dye et.al. - 88] J. Dye, F. Lanzinger and C. Stansbury, "Shark2 User's Manual Version 3.0", Advanced Decision Systems, Mountain View, California, December 1988.

[Edelson et.al. - 88] D. Edelson, J. Dye, T. Esselman, M. Black, and C. McConnell, "VIEW Programmer's Manual", Advanced Decision Systems, Mountain View, California, June, 1988.

[Gelband and Lawton - 88] P. Gelband and D. Lawton, "Perceptual Grouping Network Architecture", *IEEE International Conference on Neural Networks*, Boston, Massachusetts, March, 1988.

[Hopfield and Tank - 85] J. J. Hopfield and D. W. Tank, "'Neural' Computations of Decisions in Optimization Problems", *Biol. Cybern. 52*, No. 141, 1985.

[Hopfield - 84] J. J. Hopfield, "Neurons with Graded Response have Collective Computational Properties like those of Two-State Neurons", *Proc. Natl. Acad. Sci. USA 81*, No. 3088, 1984.

[Kuipers and Levitt - 88] B. Kuipers and T. Levitt, "Navigation and Mapping in Large Scale Space", *AI Magazine*, July, 1988.

[Lawton and McConnell - 88] D. Lawton and C. McConnell, "Image Understanding Environments", *IEEE Proceedings*, August, 1988.

[Lawton and McConnell - 87] D. Lawton and C. McConnell, "Perceptual Organization Using Interestingness", *Proceedings of the AAAI Workshop on Spatial Reasoning and Multisensor Fusion*, Morgan Kaufmann Publishers, Los Altos, California, 1987.

[Lawton et.al. - 88] D. Lawton, T. Levitt, C. McConnell, D. Edelson, K. Koitzsch, J. Dye, P. Kahn, P. Gelband, T. Binford, D. Chelberg, D. Kriegman, J. Ponce, H. Lim, G. Healey, and R. Vistnes, "Knowledge-Based Vision Techniques Task B: Terrain and Object Modeling Recognition - Annual Technical Report", Advanced Decision Systems, Mountain View, California, 1988.

[Lawton et.al. - 87] D. Lawton, T. Levitt, C. McConnell, P. Nelson, and J. Glicksman. "Environmental Modeling and Recognition for an Autonomous Land Vehicle", *Proceedings of the DARPA Image Understanding Workshop*, 1987.

[Lawton et.al. - 87] D. Lawton, T. Levitt, C. McConnell, P. Nelson, M. Black, D. Edelson, K. Koitzsch, J. Dye, T. Binford, D. Chelberg, D. Kriegman, and J. Ponce, "Knowledge-Based Vision Techniques Task B: Terrain and Object Modeling Recognition - Annual Technical Report", Advanced Decision Systems, Mountain View, California, 1987.

[Lawton et.al. - 86] D. Lawton, T. Levitt, J. Glicksman, C. McConnell, T. Miltonberger, H. Muller, P. Nelson, and C. Neveu, "Knowledge-Based Vision Techniques Task B: Terrain and Object Modeling Recognition - Annual Technical Report", Advanced Decision Systems, Mountain View, California, 1986.

[Levitt and Lawton - 89] T. S. Levitt and D. T. Lawton, "Visual Re-acquisition of Geographic Locations", *AAAI 1989 Spring Symposium: Robot Navigation Symposium Working Notes*, March, 1989.

[Levitt et.al. - 87] T. Levitt, D. Lawton, D. Chelberg, P. Nelson, and J. Dye, "Visual Memory Structure for a Mobile Robot", *Proceedings of the AAAI Workshop on Spatial Reasoning and Multisensor Fusion*, Morgan Kaufmann Publishers, Los Altos, California, 1987.

[Levitt et.al. - 87] T. Levitt, D. Lawton, D. Chelberg, and P. Nelson, "Qualitative Navigation", *Proceedings of the DARPA Image Understanding Workshop*, Los Angeles, California, 1987.

[Levitt et.al. - 87] T. Levitt, D. Lawton, D. Chelberg, and P. Nelson, "Qualitative Landmark-Based Path Planning and Following", *AAAI-87 National Conference on Artificial Intelligence*, Seattle, Washington, 1987.

[Levitt - to appear] T. Levitt, D. Lawton, D. Chelberg, and K. Koitzsch, "Qualitative Navigation for Mobile Robots", *AI Journal*, to appear.

[Levitt et.al. - 87] T. Levitt, D. Lawton, D. Chelberg, P. Nelson, and J. Dye, "Visual Memory Structure for a Mobile Robot", *Proceedings of the AAAI Workshop on Spatial Reasoning and Multisensor Fusion*, Morgan Kaufmann Publishers, Los Altos, California, 1987.

[Martelli - 76] A. Martelli, "An Application of Heuristic Search Methods to Edge and Contour Detection", *Commun. ACM 19*, 2, February 1976, pp. 73-83.

[Martelli - 72] A. Martelli, "Edge Detection using Heuristic Search Methods", Technical Report 334, AI Lab, MIT, June 1975.

[McConnell et.al. - 88] C. McConnell, K. Reilly, and D. Lawton, "Powervision Manual", Advanced Decision Systems, Mountain View, California, 1988.

KNOWLEDGE-BASED VISION TECHNOLOGY

OVERVIEW FOR OBSTACLE DETECTION AND AVOIDANCE†

K.E. Olin, M.J. Daily, J.G. Harris, F.M. Vilnrotter
Hughes Research Laboratories, Artificial Intelligence Center
3011 Malibu Canyon Rd., Malibu, CA 90265

Introduction. This overview summarizes the progress of vision research at the Hughes Artificial Intelligence Center in the area of autonomous navigation of outdoor robots.

A set of experiments designed to determine the feasibility of autonomous cross-country terrain navigation were successfully executed by Hughes with the DARPA/Martin Marietta Autonomous Land Vehicle (ALV) at the Denver test site [Daily et al., 1987]. The highlights of two separate sets of experiments are shown in Figure 1. Significant technological developments and system contributions were demonstrated by these experiments: the hierarchical perception system allowing multiple levels of interaction with a planning system, the concepts of virtual sensors and behaviors for local vehicle control, a formal definition for obstacles using a three dimensional Cartesian Elevation Map and a vehicle model, and multiple algorithms for obstacle detection including the vehicle trajectory algorithm. The experiments demonstrated the feasibility of an experimental system operating with conventional computing hardware distributed both on and off-board the vehicle. Such a configuration reduces the amount of preparation otherwise required by specialized hardware and software facilities.

The experiments also validated our approach of using simulation in our lab to close the loop between sensing, acting and planning. Our simulation capabilities include terrain modeling, synthetic laser ranging from any position and orientation, vehicle control simulation, and the ability to mimic the software and hardware configuration (including the communication network) on-board the ALV. The purpose of the simulation is fourfold. First, it allows us to test the efficiency, correctness, and usefulness of the current methods. Second, it provides a realistic environment for development of new capabilities. Third, it exercises the potential interfaces and timing requirements between planning, perception, and vehicle control, such that problems are identified prior to actual vehicle experiments. Finally, it provides a demonstration of working concepts and systems. Our successful experiments and efficient vehicle schedules have proven the value of simulation. The perception software was ported from the lab to the vehicle experiments with no modifications. We continue to evaluate our system using simulation.

The cross country experiments provided us with valuable hands-on experience and exposure to the problems associated with outdoor robots. As a result, we have directed our research toward object recognition and information fusion. We have developed a method for segmentation of color imagery that simultaneously smooths and finds color discontinuities based on Markov random fields. We have experimented with techniques that fuse sequential frames of laser range data to recover all six degrees of freedom in vehicle motion. In order to analyze a complete experiment, we have developed a representation system for combining sequences of sensed data together with prestored data such as map data. This representation supports temporal integration for improved local obstacle analysis as well as information for complete mission analysis.

† Development of this system has been supported by Defense Advanced Research Projects Agency (DARPA) contract #DACA76-85-0007.

	August, 1987	December, 1987
Experiment Period	2 Weeks	1 Week
Maximum Distance	200 Meters	735 Meters
Fastest Speed	1km/Hr	3.5 Km/Hr
Perception Technology	Cem & Vehicle Model On-board ALV (Sun 3)	Same
Planning Technology	Maps & Behaviors ALV Lab (Symbolics)	Sub-goal Ellipses Improved Displays in Lab
Communication	Rf Link	Rf Link
System Limitations	Sys. Loosely Coupled Vehicle Interfaces	Terrain Vs Sensor Resolution
Major Accomplishments	First Cross Country Demonstration Map & Sensor Based Navigation Verification of Simulation	Reliability and Repeatability Validation of virtual sensors & behaviors Longer runs (30+ minutes)

Figure 1. Results of Hughes Cross Country Experiment On-board the ALV

1. Technical Overview

1.1. Problem

Operating in an unconstrained natural environment poses many problems for the perception and planning components of an autonomous system. The complexity of the environment makes it impossible to predict every relevant situation, such that the system must reason with the knowledge available and navigate accordingly. It is crucial that perception recognize critical features in a timely fashion since the value of these observations rapidly decays and new features are constantly being exposed with the vehicle motion. The vehicle planning system must make maximal use of all accessible knowledge, such as map data and expected landmarks, to help direct perceptual tasks. Thereby, specialized tasks determined by planning can impose additional constraints which make the recognition of critical features in the environment more likely. This idea of using a planning system to unify sensing and control goals is inherent in the Hughes system architecture.

Previous demonstrations on the ALV were restricted to road following problems. Navigation in an unstructured environment such as cross-country terrain presents a significantly different set of problems. In road-following applications, knowledge and expectations of the traversable surface simplifies the otherwise difficult task of processing video data to detect roads. Certain assumptions concerning surface smoothness and continuity and fairly well constrained surface properties such as width, color, and slope are exploited. Natural terrain scenes, on the other hand, are less "predictable" than man-made scenes. Seasonal changes, weather conditions, or erosion result in a constantly changing environment. The features which might be used to compose precise models of terrain objects are difficult to capture and represent in current computer vision systems. We exploited the advantages gained through the use of a range-finding sensor.

Obstacles may be defined in the most simple case for road following as any perturbation of the road surface. Planning in road following scenarios is then responsible for maintaining a course "on" the road allowing very little variation. By comparison, in cross-country navigation there is no pre-defined course for traversal; a planning system must both plan a potential course and monitor local progress along that course as execution requires frequent avoidance of obstacles. The sensed information therefore is interpreted to locate a much richer variety of objects that help in identifying obstacles and areas safe for navigation.

Obstacles in cross-country terrain are varied and plentiful. Many ad hoc definitions for obstacles have been suggested; for these definitions, nature has provided many examples and counter-examples. A more formal definition is required. Hughes has advocated defining obstacles in terms of the vehicle itself; that is, a vehicle model that represents the traversability constraints. Our current vehicle model defines obstacles in terms of suspension, clearance, and vehicle tilt/pitch.

1.2. Summary of Approach

A knowledge-based system which can effectively accumulate, represent, and disseminate diverse knowledge is being developed. The perception system must actively select and integrate sensory information. The wide variety of capabilities and applications is focused through requests for information issued by planning. Although planning requests serve as an attention-focusing mechanism, perception must be capable of satisfying multiple requests simultaneously. Since planning requests may demand different forms of data with different requirements for assimilation or immediacy of that data, perception must be capable of fusing data from multiple sensors and multiple "looks" as well as providing specialized processing for real-time control tasks.

For autonomous navigation, perception must use sensed data to determine local areas of obstacles and free space. In our approach, obstacles were formally defined as areas violating the vehicle suspension, clearance or slope tolerances, and areas where there is not enough sensed information to determine a safe path ("unknown" areas). Perception verified the safe distance along the set of potential vehicle paths requested by the planner; each path termination would be labeled as one of these obstacles. Perception could also provide a coarse measure of "quality" for each path, such as the average slope or variation in elevation along a path. Our approach to perception has been closely integrated with the vehicle planning technology being developed at Hughes [Keirsey et al., 1988]. The planner can use the information provided by perception to avoid paths terminating with known obstacles and to explore along paths with unknown areas.

Object recognition is achieved by enriching the geometric descriptions for obstacles obtained from laser range data with color data and map expectations. The objective is to distinguish obstacle from non-obstacle. For example, a common problem at the Denver test site involves the recognition of tall thistles, which the vehicle may safely traverse, as distinguished from a thin metal post of similar diameter and height, which is an obstacle. Route determination is also affected by identification of dominant terrain features. Another example from the Denver test site is the recognition of an obstacle as a gully that can then be located in the digital terrain map and more efficiently avoided or traversed.

2. Color Image Analysis

In order to effectively locate obstacles such as rocks and bushes, we have investigated the use of color imagery obtained via a RGB color video camera. Our work has focused on two main areas: analysis of color space and representations, and color image segmentation and labelling.

We have developed a method of simultaneously smoothing and finding color discontinuities. For our purposes in dealing with natural outdoor imagery we have investigated methods of calculating color differences and the corresponding strengths of this difference required for the formation of color discontinuities. Our current segmentation exploits the expected dependencies of neighboring pixels inherent in Markov random fields (MRF). Briefly, within the MRF framework, we defined two processes: 1) the line process, which governs the formation of discontinuities (consisting of horizontal and vertical components), and 2) the color process, which performs smoothing where discontinuities do not exist. By allowing the binary line processes to vary continuously between 0 and 1, the final discontinuities are formed in an iterative fashion. The energy function minimized using Hopfield nets has four contributing terms: a smoothing term, a data term, a potential energy term for the line processes, and a gain term. Our research has included studies of the influence and interaction of these parameters. A full description of this work, including *color* images showing results, is found in [Daily, 1989] in these proceedings.

3. Information Fusion

During the past year our research on recognition of natural terrain objects has emphasized fusing information. Such information may be collected or computed from multiple sensors, available as a sequence of image frames collected over time, or may be obtained from non-image sources such as digital maps. In this overview, we will describe our techniques for fusing multiple frames of data obtained from laser range scans, and the development of the Automated Topographic Terrain Information Collector (ATTIC). Computed Cartesian color maps with rectified color data and navigational preferences in the form of traversability weights will also be introduced.

3.1. Multiple Frame Fusion Techniques

Since the vertical field of view of the laser range scanner is 30 degrees, the terrain immediately in front of the sensor is not be visible. For example, the ALV scanner cannot see obstacles directly in front of the vehicle due to the height and tilt of the laser range scanner. The closest scanned ground is approximately 13 feet in front of the vehicle. One may choose to ignore that lack of data and assume an obstacle-free flat plane immediately in front of the vehicle. Another way to fill in the unknown area is to replicate the nearest scanned ground forward until it is underneath the vehicle. This method may project an obstacle up to the vehicle bumper that is actually 13 feet away and therefore block a viable avoidance route. These are naive impractical solutions, especially for cross country applications. The best approach is to fill in the unknown area using data from previous scans.

Fusion of laser range information requires exact knowledge of the vehicle displacement between successive images with vehicle motion having six degrees of freedom. Orientation sensors on board the ALV record the vehicle's heading, and the x and y displacement values. Hughes introduced an additional sensor to estimate vehicle pitch and roll. However, there is no means to measure the change in elevation between image samples. In natural terrain, elevation values may vary drastically between image samples; for instance, the experiments show elevation changes of two to three feet with an average sampling rate of eight seconds. A simple estimate of the change in vehicle elevation, z, between two scanning locations may be calculated by taking the difference of the average z value of a small patch of common ground. However, this method is severely limited by the accuracy of the local navigation system (LNS). Consequently, we have developed two more sophisticated methods for motion recovery which do not rely on accurate LNS information and can recover from an image sequence all six degrees of freedom in vehicle motion. The first algorithm, *rigid body motion recovery*, obtains all six degrees of freedom of motion assuming a static environment. The second algorithm, *range flow*, computes a dense 3D vector motion field. Our motion analysis algorithms have been done in collaboration with Berthold Horn from the MIT Artificial Intelligence Lab.

All of our range motion analysis is performed with Cartesian Elevation Maps (CEMs). The CEMs are the same three dimensional terrain representation used for single frame obstacle detection. We provide a brief description of the CEM for the reader's convenience.

3.1.1. Cartesian Elevation Map Construction

The Cartesian Elevation Map (CEM) is an alternate representation for range information in which data from the viewer centered coordinate system of a range sensor is transformed into a Cartesian, z(x,y), coordinate system [Daily, Harris, Reiser, 1988]. This results in a down-looking, map view representation of terrain. The first step in converting from the corrected, pre-processed laser range data to the CEM is to calculate the (x, y, z) Cartesian values from each range measurement. Optics in the sensor cause scan rays to diverge as they travel away from the sensor. When a divergent ray falls on an object at some arbitrary angle, it illuminates an elliptically shaped area often referred to as a laser "footprint". Distance to the footprint is a reflectance weighted average over the illuminated area. Each of the 3D points in the CEM denotes the approximate location of the center of a footprint.

Figure 3a shows the actual (x,y) positions of each one of the scanned points for the image in Figure 2 within a 64 foot by 80 foot region in front of the scanner. Elevation data (z values) are present only at these sparse points; we define these points as constraint points in later interpolation process. As one would expect, the sparsity of scanned points increases with distance from the scanner. We approximate the complex laser footprint scanning procedure as a smoothing followed by a sampling process. Theoretically, if the terrain were sampled well enough (i.e. within the Nyquist rate), then we could accurately reconstruct the original terrain by interpolating a smooth surface between scanned points. Clearly, there will always be some unknown regions in which there are not enough scanned points to accurately reconstruct a surface. For example, any region outside the field of view of the scanner or in the shadows of tall features in the terrain will be unknown. These areas must be located and excluded from the interpolation process. Figure 3b shows the regions where the density of scanned points was deemed too low to accurately reconstruct a surface.

An iterative interpolation algorithm was used to fill in a continuous surface in all regions with a sufficiently dense sampling of points. This algorithm is similar to the standard routines used for interpolating surfaces from sparse stereo data. Rather than fitting a thin plate to the surface (i.e. minimizing the quadratic variation), we have found that fitting an elastic membrane is satisfactory. Intuitively, this algorithm is equivalent to fitting a rubber sheet over the set of constraint points and finding the resultant minimal energy surface. Figure 3c depicts the final interpolated CEM, where brighter areas denote higher elevations.

Figure 2. Laser range image.

Figure 3. Cartesian Elevation Map;
a) Constraint points, b) Known areas,
c) Interpolated elevation data.

In cross-country navigation, nearly all useful information in a laser range image is contained within the first 64 feet of the scanner's field of view. With a zero offset setting on the range scanner, this represents the first ambiguity level. While we developed algorithms to process information in the subsequent ambiguity intervals, the generally poor quality of the data beyond the first interval does not warrant the investment in processing time. Our current methods do not use any data in the remainder of a column after an ambiguity jump. Though this approach works well in many scenes, it will fail in those containing overhanging objects. This is a serious problem since the current technique will entirely ignore obstacles such as tree branches, allowing the vehicle to collide with them. The CEM also poorly represents features on near vertical surfaces. If objects such as these ever dominate an environment, it will be wiser to compute motion from the original range images.

The following section discusses how the CEM concept is used to fuse data from multiple scans. The algorithms we use for motion analysis process the CEM without regard to the original range sensor, such that a stereo depth map could also be used. The CEM has also been used to aid in the combination of data from more than one sensor.

3.1.2. Rigid Body Motion Recovery

The rigid body motion recovery algorithm obtains all six degrees of freedom for a moving vehicle platform from sequential pairs of CEMs. The original range images cannot be approximately registered since they are scanned from a viewer centered coordinate system. CEMs, on the other hand, can be translated and rotated by properly moving constraint points and re-interpolating the CEM. We have also found that motion analysis is mathematically much simpler when dealing with CEMs than with the original range images. Making the restrictive assumption of a static environment allows us to replace a severely underconstrained system with an overconstrained system which has one equation at each pixel and only six unknowns in the whole system [Horn, Harris, 1989].

The method assumes that the motion between scans is small compared with the feature size of objects in the environment. For example, if two CEMs are misaligned by just a few feet, narrow obstacles such as gullies may not overlap in the two CEMs, resulting in inaccurate motion recovery. However, in our experiments and data collection, the sampling rate between CEM's was typically eight seconds or more. When vehicle motion between successive frames is large (e.g. 10 feet), two possible approaches are: 1) to use motion estimates obtained from on-board sensors to approximately align CEMs, and 2) to use a coarse to fine approach. For the first method, on-board sensors give reasonable estimates for x, y, and heading. Using this information, the rigid body motion recovery algorithm computes all six parameters between two successive frames fairly accurately. We have found, however, that the roll of the vehicle can vary dramatically in natural terrain so that there is an unacceptable accumulation of error acquired from the fusion of many frames (over 100 frame sequences). In the latter method, on a coarser level of resolution, the CEMs will be heavily smoothed so that only large features such as the rock outcrops and the gradual curvature of the ground remain. The motion computation at this scale will be very robust to misalignments (because of the large feature sizes) but the precision of the computation will be low (again because of the large feature sizes). The coarse scale computation of motion should be a good enough estimate for the next finer level of resolution to compute a more accurate motion, and so on until the desired resolution is achieved.

Figure 4 shows a sequence of range images, corresponding CEMs and the resulting fused CEM. The rigid body motion recovery algorithm works accurately. Given only estimates of three of the six degrees of freedom (x, y and heading), the algorithm computes all six parameters. We are working on methods of quantifying how well the algorithm works, but CEMs can now be fused adequately enough to reliably operate the ALV. In fact, in one of the scans, Joe Human walked through the image and appeared in one of the CEMs. Joe's appearance seemed to have no effect on the fusing process, since the method robustly averages constraints from the total area of the CEM. We have not yet experimented with coarse to fine fusion techniques.

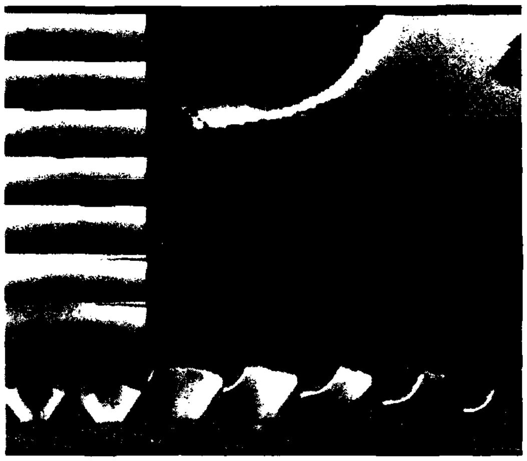


Figure 4. Rigid Body Motion Recovery
Left) Sequence of laser range images, Bottom) Corresponding CEM sequence, Center) Fused CEM

3.1.3. Range Flow

Range flow can be understood as a three dimensional extension of traditional two dimensional optical flow. In optical flow, two dimensional motion in the image plane is computed by assuming that the brightness of objects is independent of small changes in viewer rotation and translation. In range flow, we know exactly how the range of objects changes with viewer motion. We use the same motion constraint equation from rigid body motion recovery. Note that in contrast with 2D optical flow, no comparable assumptions about the change of z over time need be made. We still have an ill-posed problem since we have 3 unknowns (x, y, and z) at each pixel and only one constraint equation. Just as in optical flow, we assume that the motion fields are smooth to regularize the problem. Minimizing the resultant energy formulation yields an iterative technique for computing the dense velocity fields. Discontinuities in the velocity fields and fusion with video data should be natural extensions with use of the powerful MRF ideas developed by Poggio and his colleagues at MIT. Preliminary results for the range flow algorithm have been obtained. We plan to investigate the range flow algorithm more fully when we need to deal with dynamic environments and multiple sensor fusion. We do not need its power for the autonomous navigation scenarios we are planning in the short term.

3.2. Automatic Topographic Terrain Information Collector

Many types of autonomous vehicle missions will require a collection of data along the perceived route. In particular, it will be important to keep road, obstacle, and perhaps landmark information available for map verification or for planning a return trip. Since it may not be feasible to keep a full resolution fused CEM for the entire route lower resolution versions may be sufficient along with symbolic information gathered about specific scene objects. The scene objects of interest may be compact, e.g., rocks and trees, or extended, e.g., ravines and steeply sloping terrain. We therefore need a complex knowledge representation system that is able to integrate information perceived over time and accommodate different kinds of information, including scalar arrays and symbolic information, stored at a number of different resolutions.

The Automated Topographic Terrain Information Collector (ATTIC) is a multi-resolution knowledge representation system designed to organize information from different sources collected or computed over time. It can also be used to integrate new information with previously stored information (e.g. maps) to achieve information fusion for perceptual understanding. The ATTIC allows the user to select the number of resolutions or levels, the resolution of each level, and the row (column) dimension in meters for the arrays at each level. A single level within the ATTIC includes local terrain information and obstacle arrays as well as world location and display information. As information is collected, the ATTIC multi-resolution arrays are updated using a modular floating origin scheme to optimize the display of large amounts of data with efficient memory allocation.

3.2.1. Objectives

For autonomous vehicle navigation in complex terrain it is important to have a reliable means of interpreting the environment. Wherever possible, multiple information sources should be used to reinforce correct interpretations, evaluate possible anomalies and add to the richness of the scene object descriptions. Our goal in developing the ATTIC is to include many types of information to support all levels of information fusion. Scene objects extracted from perceived data have more complete descriptions as more types of symbolic and/or cultural information are made available. We have evaluated the ATTIC using the CEM data obtained during our 1987 ALV experiments, together with map-based cultural and elevation information available for the same area. Test data have included sequences of over 300 laser range scans containing complex vehicle paths that cross over the same area from multiple directions.

3.2.2. Current Functionality

The ATTIC system was designed to: 1) organize information from different sources collected or computed over time, and 2) integrate newly sensed information with previously stored information such as maps. The sensed information includes CEM sequences, with each CEM calculated from the laser range data and fused with previous data using the rigid body motion recovery algorithm. The system has multiple resolution levels defined by the user. Fused elevation data from CEMs is shown at two resolution levels in Figure 5.

Figure 5. Two levels of elevation data in the ATTIC;
a) Entire experiment at low resolution, b) Path cross over area at high resolution.

Other information computed as an aid in mission analysis includes a history list and a frame count array. The history list contains information about the respective CEM frames composing the current ATTIC application. The parameters in the history list consist of the CEM frame sequence number and corresponding time stamp, distance traveled since the last CEM, vehicle heading in world coordinates, current world location, current vehicle velocity, and change in elevation (delta z) when one is calculated in the ATTIC. The frame count array resides at the coarsest resolution level of the ATTIC and retains the number of the most current CEM frame adding information to that pixel's area. The frame count array also doubles as a "maybe known" array, i.e., an array in which a pixel is greater than zero if at least part of that pixel's area is known. The value of a pixel in this array can be used to index into the CEM history list. In the event that an important image feature is extracted, this index provides valuable time tags over the feature's area.

An important feature of the ATTIC is the ability to relate sensed information to a prestored digital map. Map data and map manipulation procedures are used to provide information concerning elevation and landcover as well as landform and gully locations. We have used corresponding map data for a 320 by 320 meter area representing the ALV experimentation area. Some of the cultural maps for this area are illustrated in Figure 6. In addition, a map route array was defined to display the predicted elevation values expected to overlap with the sensed data. Figure 7 shows map elevation data for the sensed elevation data in Figure 5. Therefore, as the CEMs are calculated, fused and added to the ATTIC sensed elevation arrays, a corresponding array of map information is available. Comparison of these arrays enables us to measure the relative correctness of the calculated vehicle location and elevation values. Discrepancies have been experienced in cases with either an incorrect estimate of initial vehicle roll or roll estimates were not available (or poorly measured) for each range image.

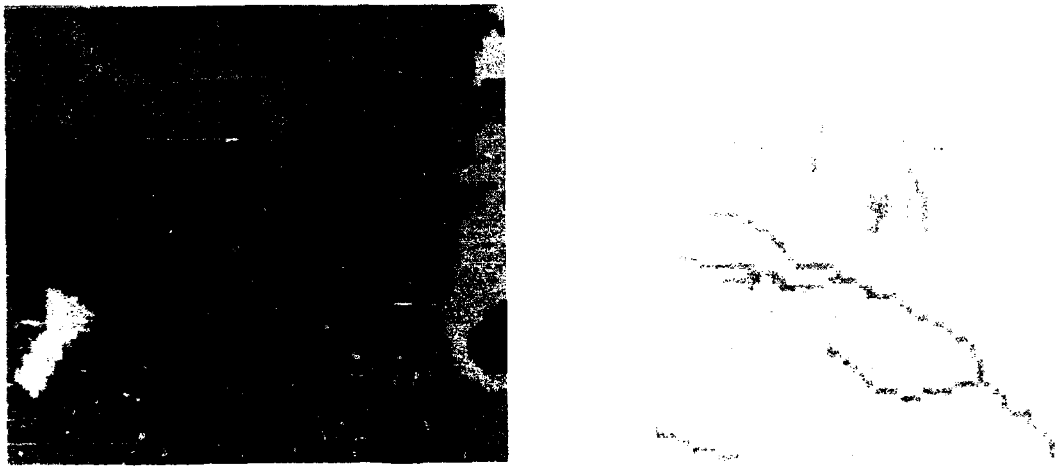


Figure 6. Cultural maps of experimental area.

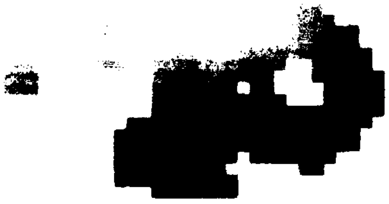


Figure 7. Predicted elevation data for vehicle path.

Figure 8. Map data overlaid with sensed terrain data.

We have also experimented with the use of the cultural terrain information for obstacle identification. In Figure 8, cultural features are mapped onto the sensed elevation data in Figure 5. The gullies along the lower diagonal of the image align well with the lower boundary of the vehicle path. However, in the upper portion of the path loop, it appears that the vehicle traveled through gullies. Two explanations are possible for this condition. First, map information does not include the depth of a gully, such that a shallow gully is safely traversed. Secondly, there may be errors in the map data. In this example, the gullies are displaced in the map; they are physically located just above the path and were successfully avoided by the vehicle in this experiment.

In order to accommodate obstacles, the slope image and obstacle mask resulting from the Gradient of Gaussian (GOG) computation [Daily, Harris, Reiser, 1988] were added to each ATTIC level. The GOG obstacle detection method was developed to calculate an approximate tertiary map of obstacles, unknown areas, and free space. Just as in conventional edge detection in video images, the CEM is convolved with an appropriately sized Gaussian mask and then the magnitude of the gradient is thresholded. The threshold value chosen corresponds to the maximum slope that the vehicle model can traverse (i.e., approximately 18°). In Figure 9 we show the GOG mask for obstacles in both low and high resolution levels of the ATTIC. The gully area shown at the bottom of the route is clearly distinguished as an obstacle (bright intensity) in the GOG image.

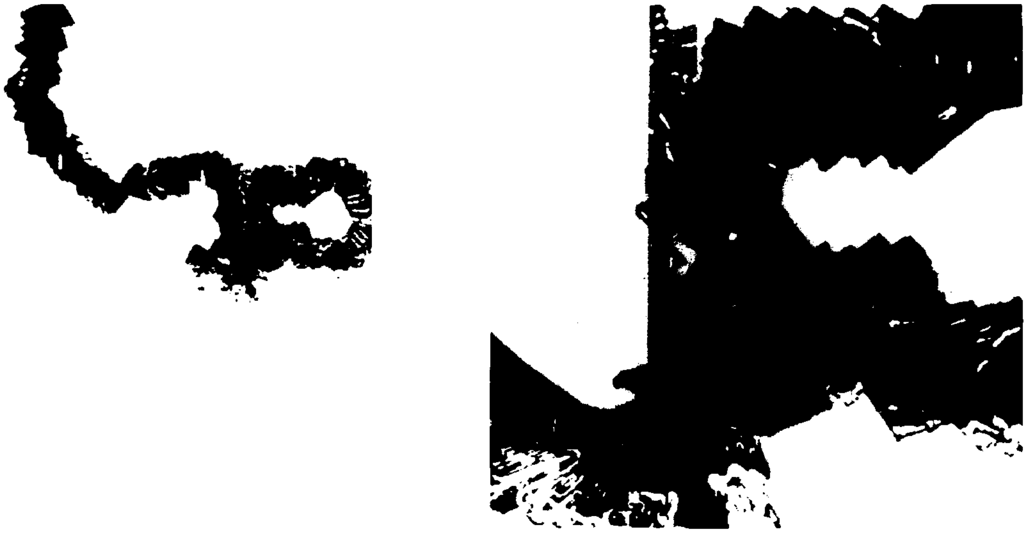


Figure 9. Obstacle detection using gradient of Gaussian algorithm.

Other information we have found useful for obstacle detection includes the number of times a pixel has been scanned ("known" pixel) and the number of times an obstacle has been detected. This information allows us to filter out obstacles detected due to spurious returns from the laser range scanner (real range data tends to be noisy). For the vehicle speeds and sampling rates for the ALV experiments, typical obstacles are seen at least three times. This type of obstacle filtering scheme tends to lose obstacles at the edges of the path since these pixels appear in fewer scans. An adequate threshold for this information should be experimentally determined.

3.2.3. Additional Maps to Augment ATTIC

It is intended that other information will be added to the ATTIC. Two fused maps that we have available are the Cartesian Weight Map (CWM) and the Cartesian Color Map (CCM). Like all of our Cartesian maps, these are downward looking maps of the local vehicle area.

The CWM was developed to provide an improved understanding of the vehicle's local environment. Each location in the CWM contains one or more weights signifying the cost of traversing through that area. Weights may be combined from diverse and independent sources such as color sensors or range scanners, virtual sensors, or processing techniques like the gradient of Gaussian (GOG) slope algorithm. The weights need not be fixed, but may be a function of terrain type, vehicle speed, and mission objectives. In this way, the CWM adds structure to potentially conflicting and diverse information about traversability. It also provides a natural way to feed a much richer global description to the local planning unit. For instance, areas in the CWM that are bumpy are penalized while smooth regions are rewarded. The CWM has been exercised with a least cost path planning algorithm in our simulation environment. It has successfully kept the vehicle safely away from obstacles, navigated a narrow corridor between two obstacles, and avoided cul de sacs which are within the field of view.

The CCM is another downward looking map, but in this case we store color information obtained from the color sensor at each map location. This results in a true rectified view of the color image without using flat ground assumptions. In our approach, the elevation map (CEM) is back projected to find the color of each pixel. In this way, no interpolation of color is necessary and as many features as possible from the video image are present in the CCM. We plan to use the CCM in conjunction with the CWM to locate obstacles that are difficult to find using only geometric information in the CEM.

4. Conclusions

Advances over the past four years in the vision and planning technologies for mobile robots have culminated in an exciting series of demonstrations. On-road and off-road experiments with both the ALV and Navlab testbeds have shown the feasibility of the current technology and systems to navigate in real world outdoor environments. The next step, the ability of a system to understand the sensed scene and to plan the vehicle actions needed to accomplish scenario goals, is beginning to reach the maturity level necessary to spawn a new series of experiments.

We are proud of our accomplishments in cross-country autonomous navigation and will continue to study issues related to mobile robots. Finally, we regret that the ALV will no longer be as a testbed for autonomous vehicle technology.

Acknowledgements

The progress presented in this overview represents the effort of the technical staff of the Computer Vision Section of the AI Center at the Hughes Research Laboratories. We would like to thank Dr. Ram Nevatia for his consultation with a number of the algorithms discussed here. We thank the Martin Marietta team for supporting the experimentation schedule and our successful cross-country demonstrations, thereby exposing us to an even greater number of interesting navigation and perception problems for continued research. Lastly, we thank Dr. David Tseng, manager of the AI Center, for his continued support of our technical pursuits.

References

M. Daily, J. Harris, D. Keirsey, K. Olin, D. Payton, K. Reiser, J. Rosenblatt, D. Tseng, and V. Wong, "Autonomous Cross-Country Navigation with the ALV," Proceedings DARPA Planning Workshop, Texas, December, 1987.

M.J. Daily, J.G. Harris, K. Reiser, "An Operational Perception System for Cross Country Navigation," Proceedings Image Understanding Workshop, Boston, MA, April, 1988.

M.J. Daily, "Color Image Segmentation Using Markov Random Fields," Proceedings Image Understanding Workshop, Palo Alto, CA, May, 1989. Also to appear in CVPR, June, 1989.

B.K.P. Horn, J.G. Harris, "Rigid Body Motion from Range Image Sequences," submitted to CVGIP.

D.M. Keirsey, D.W. Payton, J.K. Rosenblatt, "Autonomous Navigation in Cross Country Terrain," Proceedings Image Understanding Workshop, Boston, MA, April, 1988.

Image Understanding Research at GE [1]

N.R. Corby

GE Corporate Research and Development Center
Schenectady, NY 12301

INTRODUCTION

The basic paradigm of model-driven image understanding continues to be an effective one for a large number of problems encountered in machine vision. Many of our concerns in modelling have arisen from considerations of basic questions in model-driven object recognition. We have taken the view that formal geometric reasoning techniques may provide a way to address the goals of developing new, more powerful representations on which to base recognition-oriented models and new approaches to performing the recognition task.

We have attempted to progress along a dual course which seeks to demonstrate the applicability of our techniques to concrete problems such as the automated interpretation of military reconnaissance imagery, while maintaining a predominantly research-oriented focus. This dual course has not proved a hinderance and, in fact, has been beneficial in identifying specific and general problems in automated recognition. Application issues such as computational complexity and speed issues led to the vertex-pair approach as a compact feature for recognition, while the need to address the total problem of reconnaissance imagery understanding has led us towards the PACE research system described last year [Corby et al]. The results of application experiments have strongly influenced our work in more formal geometric reasoning methods. For example, the goal of automated modelling based on a combination of numerical data and algebraic constraint sets is a direct consequence of application issues, as is our research in symbolic solution methods.

This report will briefly summarize the major elements of our work. A more complete discussion can be gained by reference to the appropriate paper in this and other volumes.

MODEL-BASED OBJECT RECOGNITION

Our approach has focussed on the identification of 3-D and corresponding 2-D geometric features sufficient to recover object position and orientation under the conditions of affine projection. The object models that we use are solid polyhedral models. The 3-D/2-D feature that we have developed is the "vertex-pair" [Thompson and Mundy 88]. We use a transform space "voting" proceedure to exhaustively compare all 2-D features to a specific 3-D feature, casting a set of votes for each comparison. Sucessive consideration of a small number of 3-D vertex-pairs results in clusters in transform space, since all 3-D features are taken from a rigid-body solid model. Recognition is acheived by detecting clusters in transform space and verifying the appropriateness of detected clusters (hypothesis verification). The transform space approach has allowed us to progress in spite of the sometimes erratic and fragmented performance of current segmentation algorithms.

Two major areas of interest during the past year have been automated methods to select 3-D matching features and automated methods to verify match hypotheses. In work reported last year [Mundy et al88]. we described a feature assessment metric and initial work in using the metric to select minimal sets of 3-D features. We are continuing our efforts in that area. Using polyhedral models for some aircraft, the vertex selection software considered all possible vertex pairs and selected a mimimal set. The results of using these selected vertex-pairs agreed very closely to hand selected vertex-pair sets. An equally significant area that we have devoted our efforts to this year is automated methods for hypothesis verification. Two papers in this workshop deal with the topic, [Heller and Stenstrom] and [Thompson]. The first describes our investigation of methods to automatically determine whether a hypothsized recognition instance is in fact supported by the underlying image data. In the past our verification techniques have been largely manual and rely on the operator to establish plausibility. The need to automatically verify correct hypotheses (and reject

[1] Work at GE was supported in part by the DARPA Strategic Computing Vision Program in conjunction with the Army Engineer Topographic Laboratories under Contract No. DACA76-86-C-0007 and the Air Force Office of Scientific Research under contract No. F49620-89-C-0033.

incorrect hypotheses) is important for another reason. Due to errors in segmenting the original image or to inaccuracies in forming 2-D features or perspective transformation effects, it is often the case that we need to use fairly loosely specified match detection criteria. We would like to be able to select likely candidates from this larger, more loosely specified set and then improve the accuracy of recognition. In [Thompson], we describe iterative procedures to improve or refine transforms. Automated hypothesis verification routines are required to select likely candidates and to control the iterative processes.

The modeling efforts reported on last year have continued. We have continued to expand the capabilities of our automated modeller. Two examples that we have created include a model of a Blackjack bomber (with approximately 50 faces) and a model of a prototype Mars ascent vehicle (with approximately 200 faces). Both models were created from outline drawings and luminance images in a largely automated fashion.

GEOMETRIC REASONING PROGRESS

Much of our past work in geometric reasoning has been directed towards the problem of automated methods for model construction. Since our basic approach to object recognition has been largely model-driven, we have necessarily pursued research into geometric model formation. Work we have described in the past featured the use of multiple modalities such as luminance and range data as well as luminance methods that use Boolean intersection techniques to build models.

A second approach to automatically producing models was to use geometric reasoning techniques to establish a set of constraints for an object that would capture the geometry of the object from projections (photographs) of the object. In the process of refining approaches to this goal, much work has been done on identifying the nature of the constraint equations and in identifying methods that can be applied to the solution of the resulting constraint equation sets.

A new direction in the application and development of geometric reasoning techniques has been initiated. Motivated by our work in object recognition and scene modeling for photointerpretation [Corby et al], we have begun work on the synthesis of symbolic and numerical methods for solving systems of geometric constraint equations. Object models and the configuration of such objects in the world can be represented as a set of algebraic equations and inequalities which represent the constraints imposed by object geometry and by cartographic data. The advantage of representing objects and environments in terms of constraints is that multiple sources of information can be integrated in a uniform fashion. The central problem is to develop reliable and efficient methods for solving the constraint equations.

We have carried out modeling experiments with a nonlinear programming package available under the IMSL library. The results are described elsewhere in these proceedings [Mundy and Vrobel]. It is clear that a purely numerical approach does not provide adequate robustness. Consequently, we have begun the development of a hybrid approach where symbolic geometric reasoning techniques are used to define the singularities and poor convergence regions of the constraint manifold. An algorithm has been developed for determining the free parameters of a model structure. An algebraic technique has also been developed for determining the singularities of the jacobian of the constraint equations [Mundy and Vrobel]. The final goal is to be able to automatically generate robust numerical algorithms from a symbolic specification of the model and environment geometry.

Our effort in the development of efficient techniques for reasoning about nonlinear inequalities is continuing. The research is directed at the problem of matching parametric object models to image features. The errors in image segmentation and model definition are represented as tolerance inequalities bounding the measured features. We have completed our study of SUP-INF algorithm and the use of the Groebner basis to rewrite the terms of the inequalities in conjunction with a system of equations [Final Report].

A new approach is being explored which converts a system of nonlinear inequalities into a linear form by abstracting each nonlinear term into a single variable [Cyrluk and Kapur]. The linear system is solved using the simplex or SUP-INF algorithms. The resulting bounds on the nonlinear term are then propagated inward to bound individual variables. The initial results are quite encouraging and should be applicable to other AI applications, such as qualitative reasoning.

References

[Thompson and Mundy87] Thompson, D., Mundy, J.L., "Three-Dimensional Model Matching From an Unconstrained Viewpoint", Proc. IEEE Robotics and Automation, 1987, pp. 280.

[Thompson and Mundy 88] Thompson, D.W. and J.L. Mundy, "Model-Based Motion Analysis", Proc. 4th International Symposium on Robotics Research, MIT Press, 1988.

[Mundy et al88] Mundy, J., Heller, A. and Thompson, D., " The Concept of an Effective Viewpoint", Proc. DARPA Image Understanding Workshop 1988.

[Corby et al] Corby, N.R., Mundy, J.L., Vrobel, P., "PACE-An Environment For Intelligence Analysis", Proc. DARPA Image Understanding Workshop 1988.

[Thompson] Thompson, D.W., "Edge Based Transform Refinement", Proc. DARPA Image Understanding Workshop 1989.

[Heller and Stenstrom] Heller, A. and Stenstrom, R., "Verification of Recognition and Alignment Hypotheses By Means of Edge Verification Statistics", Proc. DARPA Image Understanding Workshop 1989.

[Cyrluk and Kapur] Cyrluk, D. and Kapur, D., "Reasoning About Nonlinear Inequality Constraints: A Multi-level Approach", Proc. DARPA Image Understanding Workshop 1989.

[Mundy and Vrobel] Mundy, J. and Vrobel, P., "Constraint Based Modeling", Proc. DARPA Image Understanding Workshop 1989.

[Final Report] Corby, N.R., Mundy, J.L., Kapur, D., DARPA Final Report No. DACA76-86-C-0007, submitted to US Army Engineer Topographic Laboratories, Fort Belvoir, VA February 1989.

UNDERSTANDING SCENE DYNAMICS

Bir Bhanu

Honeywell Systems and Research Center
3660 Technology Drive
Minneapolis, MN 55418

ABSTRACT

The objective of our work in understanding scene dynamics is to develop robust techniques for target tracking and recognition from a moving robotic vehicle. The topics currently under investigation are: decomposition of complex vehicle motion; qualitative 3-D scene modeling; target motion detection and tracking; map-based target tracking; inertial sensor integrated obstacle detection; adaptive segmentation; 3-D target model acquisition and refinement; landmark recognition; and terrain interpretation. This paper summarizes the progress made in each of these areas during the period from February 1988 to March 1989. We also present a brief discussion on scientific experiments, machine learning for target recognition, and scientific performance evaluation of vision algorithms and systems.

1. INTRODUCTION

This paper provides an overview of the research performed by our group during the past year. Our research in understanding scene dynamics is directed towards knowledge-based interpretation of scene dynamics and model-based target recognition. The key accomplishments of our work during the past year are: We have developed qualitative reasoning and "3-1/2-D" modeling techniques for detecting and tracking moving targets from a mobile platform in simple curved road scenes. The concept of fuzzy focus of expansion, which allows a very accurate determination of the instantaneous direction of a moving vehicle and camera rotations along the two axes (pan and tilt only), has been demonstrated. We have also demonstrated the "dynamic model matching" concept for landmark recognition, where the model generation and matching process dynamically changes as a function of range to the landmark and perspective as viewed by a mobile platform. In addition, we have performed initial experiments in digital map integrated target tracking.

We have investigated the following major topics:

(1) *Qualitative* motion detection and tracking,

(2) *Inertial sensor integrated* motion analysis,

(3) *Machine learning* for adaptive segmentation, target model acquisition, and target model refinement,

(4) *Dynamic model matching* for landmark recognition, and

(5) *Hierarchical symbolic grouping* for interpretation of terrain.

The synopsis of the technical achievements in each of these technical areas is given below. We also present a brief discussion on scientific experiments, machine learning for target recognition, and scientific performance evaluation of vision algorithms and systems.

One of the primary objectives of scientific performance evaluation is the establishment of a national research database of computer vision imagery. This database will be maintained in locations accessible to all members of DARPA's IU community through a set of uniform access procedures. Another objective is the standardization of terminology, benchmarks, and a characterization of the computer vision research infrastructure. Also, we will define a set of techniques and models for algorithm/system performance evaluation of selected matured vision algorithms.

2. QUALITATIVE MOTION DETECTION AND TRACKING

We have developed a unique approach called DRIVE (Dynamic Reasoning from Integrated Visual Evidence) based on *qualitative reasoning and modeling* for target motion detection and tracking.[3,4,14,15,17,18] The DRIVE system performs dynamic scene understanding needed to support the application of smart weapons and autonomous navigation of robotic vehicles. Instead of refining a single quantitative description of the observed environment over time, multiple qualitative interpretations of the scene are maintained simultaneously. This technique offers considerable flexibility over traditional numerical techniques which are often ill-conditioned and noise-sensitive. With DRIVE, an autonomous system can (i) detect and classify moving objects in the scene, (ii) estimate the vehicle's egomotion, and (iii) derive the 3D structure of the stationary environment.

The 3-D motion of targets is obtained from displacement vectors of point features without any knowledge about the underlying 3-D structure, discovering inconsistencies between the current state of the qualitative 3-D scene model and the changes actually observed in the scene, and by detecting moving edges and regions.[6, 13]

DRIVE uses a new algorithm for computing the region of possible focus-of-expansion (FOE) locations in image sequences, called the fuzzy FOE.[16, 17] This computation is accomplished in a unique manner by separating the rotational and translational components of the vehicle's motion and using a robust method for computing the displacement vector between two images using adaptive windows.[13]

The 'fuzzy' FOE algorithm allows the direction of instantaneous heading of an autonomous land vehicle to be precisely determined within 1° using image information exclusively. The results obtained using ALV imagery taken at five different sites demonstrate the algorithm's performance capabilities. This result has significant scientific importance for targeting applications. It allows the determination of self motion of moving imaging devices. Rotation in the horizontal and vertical directions (pan and tilt only) of ± 5° or larger can be successfully handled by the algorithm.[5] Moreover, it allows the use of *passive approaches* for surveillance activities that must detect and track moving targets and must detect and avoid obstacles using passive sensors mounted on a mobile platform.

Experiments have been carried out on 262 frames of ALV data taken at 5 different sites. Figures 1 and 2 illustrate the results.

Figure 1 shows the original image with the interesting points after edge detection and computation of the focus of expansion. Figure 2 shows the qualitative reasoning and modeling process. There are two cars in the images, one approaching the vehicle and the other receding from the vehicle. Both of the moving cars have been detected. The reasoning process is based on the changes in the expansion pattern and uses a camera model.

We have developed preliminary algorithms to integrate the DRIVE system with digital terrain elevation and land cover data. These algorithms provide information about the map location of the moving targets, the road label on which the targets are possibly traveling, and neighboring landmarks. Such information is desired for military applications and we have performed initial experiments to establish its usefulness in detecting moving targets in both high clutter and low contrast situations. Figure 3 shows an example of target detection under low contrast and high clutter situation. Target map location, the road segment on which the car is traveling, and nearby landmarks have also been detected by the algorithms which integrate map data with the motion algorithm suite.

The paper by Bhanu et. al.[13] provides details of the interest point selection, disparity analysis, fuzzy FOE, qualitative scene model, map-based tracking, and edge/region based approaches.

3. INERTIAL SENSOR INTEGRATED MOTION ANALYSIS

Land navigation requires a vehicle to steer clear of trees, rocks, and man-made obstacles in the vehicle's path while vehicles in flight, such as rotorcraft, must avoid antennas, towers, poles, fences, tree branches, and wires strung across the flight path. Automatic detection of these obstacles by passive sensors and the necessary guidance and control actions triggered by such detection would facilitate autonomous vehicle navigation.

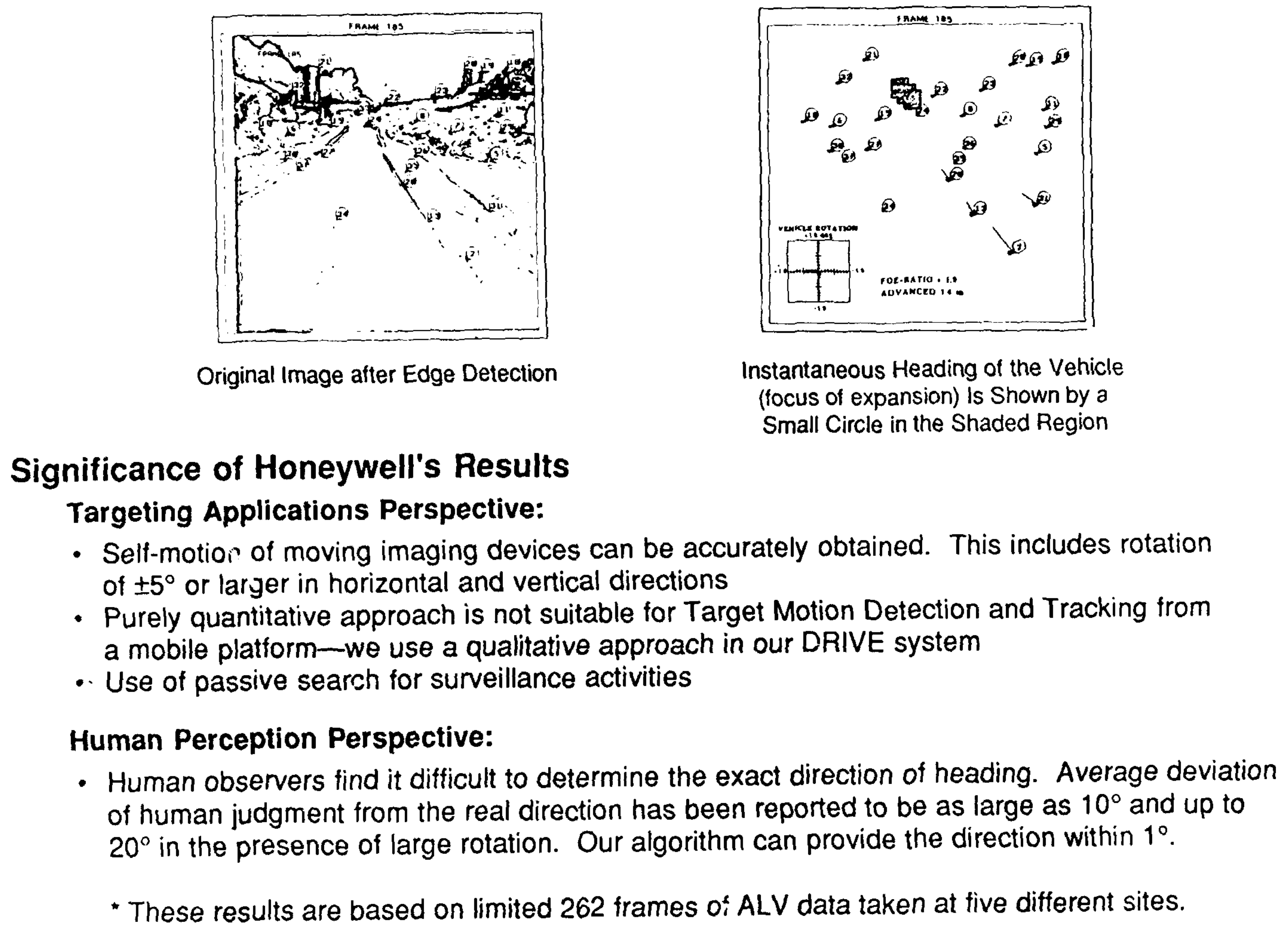
Many types of existing vehicles contain inertial navigation systems (INS) which can be utilized to greatly improve the performance of several computer vision applications such as obstacle detection, target motion detection, target tracking, stereo, etc. and make them useful for practical military and civilian applications. As an example, motion analysis techniques can effectively use the output of an INS to improve interest point selection, matching of the interest points, and the subsequent motion detection, tracking, and obstacle detection.

We are using INS measurements to enhance the quality and robustness of motion analysis techniques for obstacle detection and thereby providing vehicles with new functionality and capabilities. Details of the INS integrated motion analysis for obstacle detection are given in the paper and reports by Bhanu, Roberts, and Ming.[9, 10]

4. MACHINE LEARNING FOR ADAPTIVE SEGMENTATION AND TARGET MODEL ACQUISITION/REFINEMENT

4.1 Adaptive Segmentation Using Genetic Algorithms

Image segmentation is typically the first, and most difficult, task of any automated image understanding process. All subsequent interpretation tasks, including feature extraction, object detection, and object recognition, rely heavily on the quality of the segmentation process. Despite the large number of segmentation techniques presently available,[1, 19] no general methods have been found which perform adequately across a diverse set of imagery. Only after numerous modifications to an algorithm's control parameter set can any current method be used to process the wide diversity of



Original Image after Edge Detection

Instantaneous Heading of the Vehicle (focus of expansion) Is Shown by a Small Circle in the Shaded Region

Significance of Honeywell's Results

Targeting Applications Perspective:

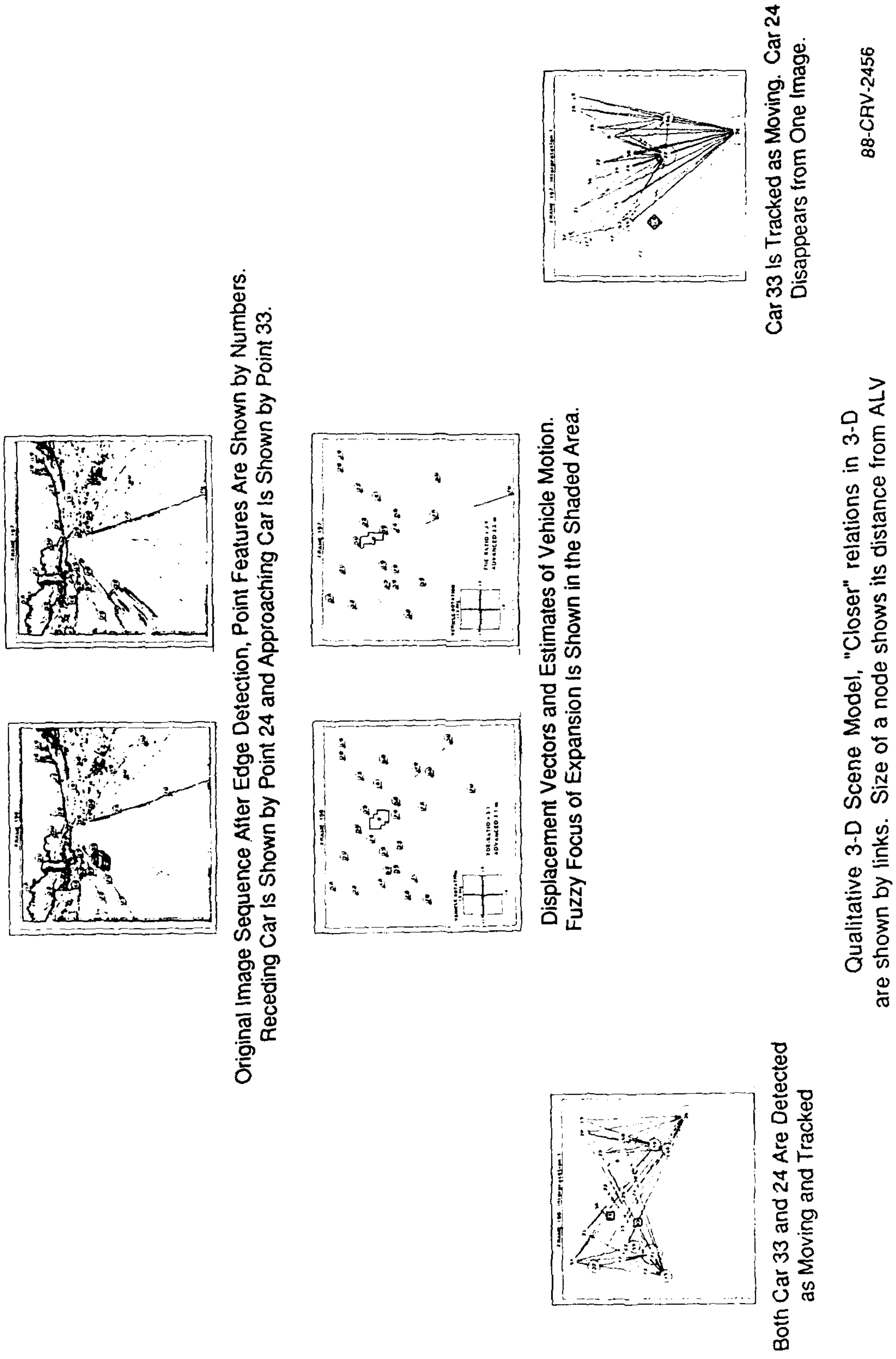
- Self-motior of moving imaging devices can be accurately obtained. This includes rotation of ±5° or larger in horizontal and vertical directions
- Purely quantitative approach is not suitable for Target Motion Detection and Tracking from a mobile platform—we use a qualitative approach in our DRIVE system
- Use of passive search for surveillance activities

Human Perception Perspective:

- Human observers find it difficult to determine the exact direction of heading. Average deviation of human judgment from the real direction has been reported to be as large as 10° and up to 20° in the presence of large rotation. Our algorithm can provide the direction within 1°.

* These results are based on limited 262 frames of ALV data taken at five different sites.

Figure 1: Direction of instantaneous heading for a moving platform is precisely given within 1°, which is a greater accuracy than human visual performance.



Original Image Sequence After Edge Detection, Point Features Are Shown by Numbers. Receding Car Is Shown by Point 24 and Approaching Car Is Shown by Point 33.

Displacement Vectors and Estimates of Vehicle Motion. Fuzzy Focus of Expansion Is Shown in the Shaded Area.

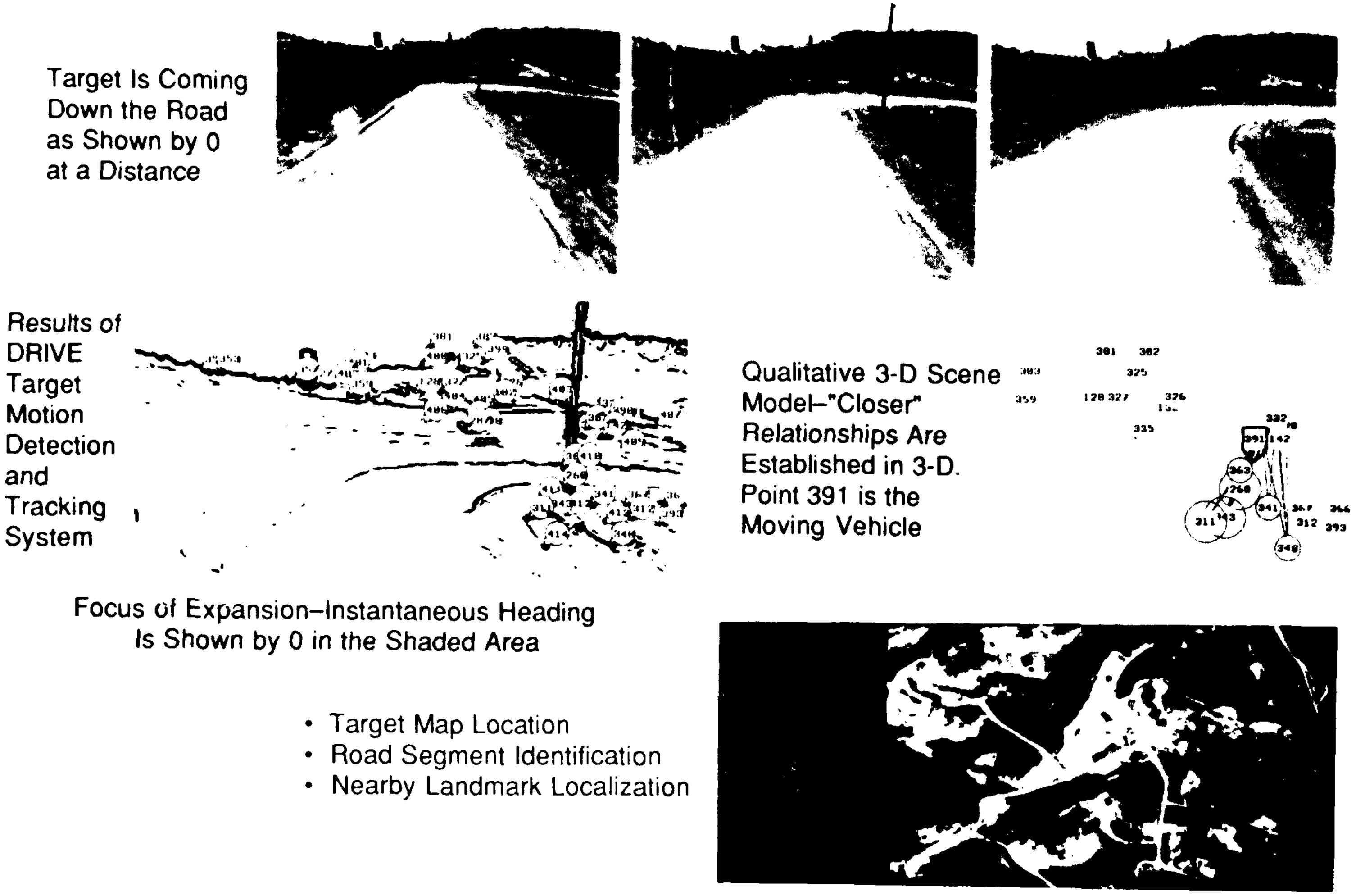
Both Car 33 and 24 Are Detected as Moving and Tracked

Qualitative 3-D Scene Model, "Closer" relations in 3-D are shown by links. Size of a node shows its distance from ALV

Car 33 Is Tracked as Moving. Car 24 Disappears from One Image.

88-CRV-2456

Figure 2: Qualitative reasoning and scene modeling provides good target motion detection and tracking capabilities from a mobile platform.



Target Is Coming Down the Road as Shown by 0 at a Distance

Results of DRIVE Target Motion Detection and Tracking System

Qualitative 3-D Scene Model–"Closer" Relationships Are Established in 3-D. Point 391 is the Moving Vehicle

Focus of Expansion–Instantaneous Heading Is Shown by 0 in the Shaded Area

- Target Map Location
- Road Segment Identification
- Nearby Landmark Localization

Figure 3: Digital map information is used to track moving targets under high clutter and low contrast scenarios.

images encountered in dynamic outdoor applications such as the operation of an autonomous robotic land/air vehicle, automatic target recognizer, or a photointerpretation task.

The image segmentation problem can be characterized by several factors which make parameter selection process very difficult. These factors include numerous control parameters, lack of segmentation model, and problems associated with the evaluation of segmentation.

We are using a machine learning technique known as a *genetic algorithm*, to perform adaptive segmentation in a closed loop feedback system. Genetic algorithms allow the segmentation process to adapt to changes in image characteristics caused by variable environmental conditions such as time of day, time of year, clouds, rain, etc. The genetic algorithm efficiently searches the enormous hyperspace of segmentation parameter combinations using a collection of search points known as a population. By combining high performance members of the current population to produce better parameter combinations, the genetic algorithm is able to locate the parameter set which maximizes the segmentation quality criteria. The paper by Bhanu, Lee, and Ming[7] provides details of the adaptive image segmentation process.

4.2 Target Model Acquisition and Refinement

A major technology gap in state-of-the-art model-based object recognition for outdoor scenes is the process of model (natural or man made) acquisition. Generally man made object models are fixed and they do not have any learning capability; therefore, they are not adequate by themselves for object recognition in dynamic environments.

Due to recent advances in machine learning technology, some of the problems encountered in the target recognition domain seem to be resolvable. Learning allows an intelligent recognition system to use situation context in order to understand images. This context, as defined in a machine learning scenario, consists of a collected body of background knowledge as well as environmental observations which may impact the processing of the scene.

Machine learning facilitates two main breakthroughs in the target recognition domain: automatic knowledge base acquisition and continuous knowledge base refinement. The use of learning in the knowledge base construction will save the user from spending the enormous amount of time necessary to derive target models and databases. Knowledge base refinement can then be incorporated to make any necessary changes to improve the performance of the recognition system. These two modifications alone will serve to significantly advance the present abilities of most target recognition applications.

We are developing a TRIPLE: *Target Recognition Incorporating Positive Learning Expertise* system for automated model acquisition and refinement. The system uses a multi-strategy technique; two powerful learning methodologies are combined with a knowledge-based matching technique to provide robust target recognition. Using dynamic models, TRIPLE can recognize targets present in the database. If necessary, the models can be refined if errors are found in the representation. Additionally, TRIPLE can automatically store a new target model and recall it when that target is encountered again. Finally, since TRIPLE uses qualitative data structures to represent targets, it can overcome problems such as image noise and occlusion.

The two main learning components of the TRIPLE system are Explanation-Based Learning (EBL) and Structured Conceptual Clustering (SCC). Explanation-based learning provides the ability to build a generalized description of a target class using only one example of that class. Structured conceptual clustering allows the recognition system to classify a target based on simple, conceptual descriptions rather than using a predetermined, numerical measure of similarity. While neither method, used separately, would provide substantial benefits to a target recognition system, they can be combined to offer a consolidated technique which employs the best features of each method and is very robust. The paper by Bhanu and Ming[8] provides more details of the TRIPLE system for target model acquisition and refinement.

5. DYNAMIC MODEL MATCHING FOR LANDMARK RECOGNITION

We have developed a technique called PREACTE (Perception REAsoning ACTion and Expectation) based on *dynamic model matching* for landmark recognition from a mobile platform.[20,21] The technique can recognize landmarks and other objects from partial and complete views in dynamic scenarios. It relies on the generation of multiple landmark descriptions from 3D models based on different estimated ranges and aspect angles. These descriptions are a result of feature, spatial, and geometric models of a single landmark. Expectations about the landmarks (appearances) vary dynamically as the autonomous robot approaches the landmark. Dynamic Model Matching also includes the generation of specific landmark recognition planning strategies whereby different features of different landmarks are emphasized at varying ranges. It is an expectation driven, knowledge-based approach and uses limited map information for updating the ALV's location in the map.

Figure 4 illustrates an example of dynamic model matching for landmark recognition from a mobile platform. Note that landmark recognition allows the determination of the ALV's position within 3 feet compared to an inertial position error of 105 feet over a distance of one mile. Figures 5 and 6 provide three examples of landmark recognition (a yellow gate and two wooden gates) taken at two different times at the Martin Marietta test site. It is to be noted that segmentation results affect the recognition results. Being very close to a landmark does not necessarily mean that its segmentation will be better than the segmentation of the image acquired at a greater range.

6. HIERARCHICAL SYMBOLIC GROUPING FOR INTERPRETATION OF TERRAIN

An autonomous robotic vehicle must be able to navigate not only on the roads, but also through terrain in order to execute its missions of surveillance, search and rescue, and munitions deployment. To do this the vehicle must categorize the terrain regions it encounters as to the trafficability of the regions, the land cover of the regions, and region-to-map correspondence. Our approach for terrain interpretation employs a robust texture-based algorithm and a hierarchical region labeling scheme for ERIM 12 channel Multi-Spectral Scanner data. The technique, called HSGM (Hierarchical Symbolic Grouping for Multi-spectral data), is specifically designed for multi-spectral imagery, but is appropriate for other categories of imagery as well. For this approach, features used for segmentation vary from macro-scale features at the first level of the hierarchy to micro-scale features at the lowest level. Examples of labels at the macro-level are sky, forest, field, mountain, road, etc. Figure 7 shows texture gradient images, and the final region boundaries for large regions. These regions are labeled using a knowledge-based classifier.

For each succeeding level of the hierarchy, the identified regions from the previous stage are further subdivided, if appropriate, and each region's labeling is made more precise. The process continues until the last stage is reached and no further subdivision of regions from the preceding stage appears to be necessary. Examples of region labels for this level of the hierarchy are gravel road, snowberry shrub, gambel oak tree, rocky ledge, etc. Further development of the technique will employ multiple sources of *a priori* information such as land cover, terrain elevation map information, range data, seasonal information, and time of day.

Details of the HSGM technique with results and examples from real imagery are given in papers by Bhanu and Symosek.[11,12]

7. VISION-BASED TARGETING EXPERIMENTS

As discussed earlier, we have developed two key algorithm suites, called DRIVE (Dynamic Reasoning from Integrated Visual Evidence) and PREACTE (Perception, REAsoning, ACTion and Expectation). DRIVE accomplishes target motion detection and tracking while PREACTE performs landmark recognition. We plan to advance this research by performing a set of scientific experiments directed towards a practical mobility and targeting application of a robotic combat vehicle.

We plan to conduct scientific experiments in two areas: landmark recognition for path traversal and target motion detection and tracking. Two series of experiments are planned, one in each of these areas. Each experimental series begins with data collection and proceeds through progressively more difficult scenarios. The final experiments in the series will be characteristic of practical mobility scenarios for a robotic combat vehicle. For both series of experiments, the vehicle will be in continuous motion.

Landmark recognition experiments include laboratory landmark recognition tests using off road data; non-real time landmark recognition in off road traversal by the ALV; real time dynamic landmark recognition in off road traversal by the ALV; and dynamic landmark model learning with return path traversal. Motion detection and tracking experiments involve verifying motion results against land navigation data; non-real time detection of multiple moving objects while maintaining reasonable rotation components of the vehicle; real time detection of multiple moving objects; integrating ETL map data with target motion detection and tracking; and advanced experiments carried out under more difficult visual scenes involving low contrast and high clutter.

We also plan to develop a flexible software architecture and the associated software for "real time" instrumentation and evaluation of the landmark recognition and the motion detection and tracking algorithms. Some of the important aspects of this work involve defining the criteria for evaluation and acquiring, retrieving, and presenting the desired information in meaningful ways so as to provide insight into the associated vision algorithms.

8. MACHINE LEARNING FOR TARGET RECOGNITION

Present target recognition systems are unable to adapt to changes in environmental conditions, target variations, and the unexpected appearance of new targets. Each of these situations affects the appearance of the targets in the image, which in turn, degrades the overall performance of current generation recognition system.

One of the key challenges to automating the target recognition process is that of automatically responding to changes occurring in the targets seen in an image. We address this problem at every stage of the multi-level vision problem by a unique multi-strategy machine learning approach not available in any current model-based recognition system. We want to show that significant benefits can accrue through applying machine learning technology to

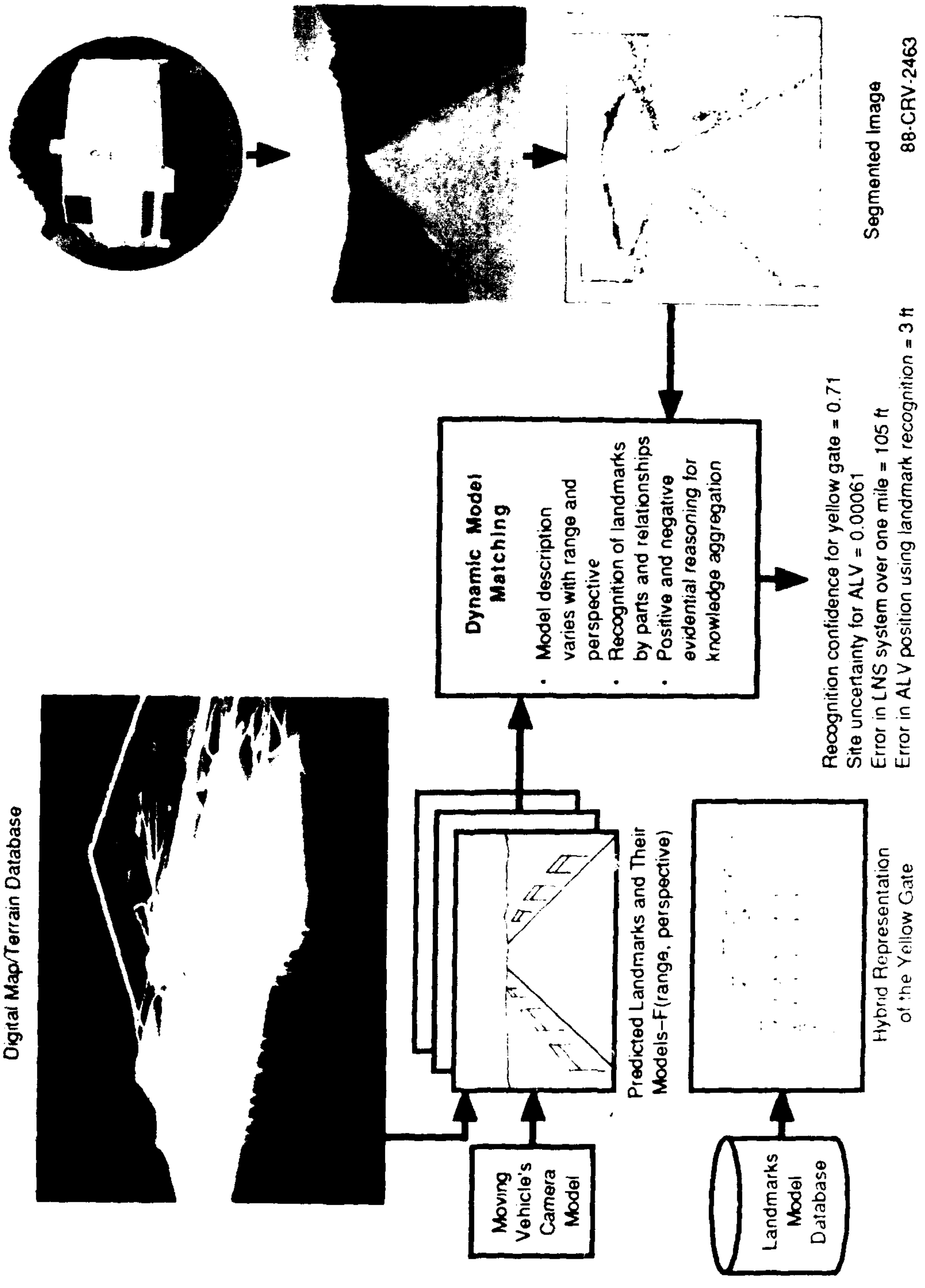

Digital Map/Terrain Database
Moving Vehicle's Camera Model
Predicted Landmarks and Their Models–F(range, perspective)
Landmarks Model Database
Hybrid Representation of the Yellow Gate
Dynamic Model Matching
• Model description varies with range and perspective
• Recognition of landmarks by parts and relationships
• Positive and negative evidential reasoning for knowledge aggregation
Recognition confidence for yellow gate = 0.71
Site uncertainty for ALV = 0.00061
Error in LNS system over one mile = 105 ft
Error in ALV position using landmark recognition = 3 ft
Segmented Image
88-CRV-2463

Figure 4: Example of Dynamic Model Matching for landmark recognition from a mobile platform.

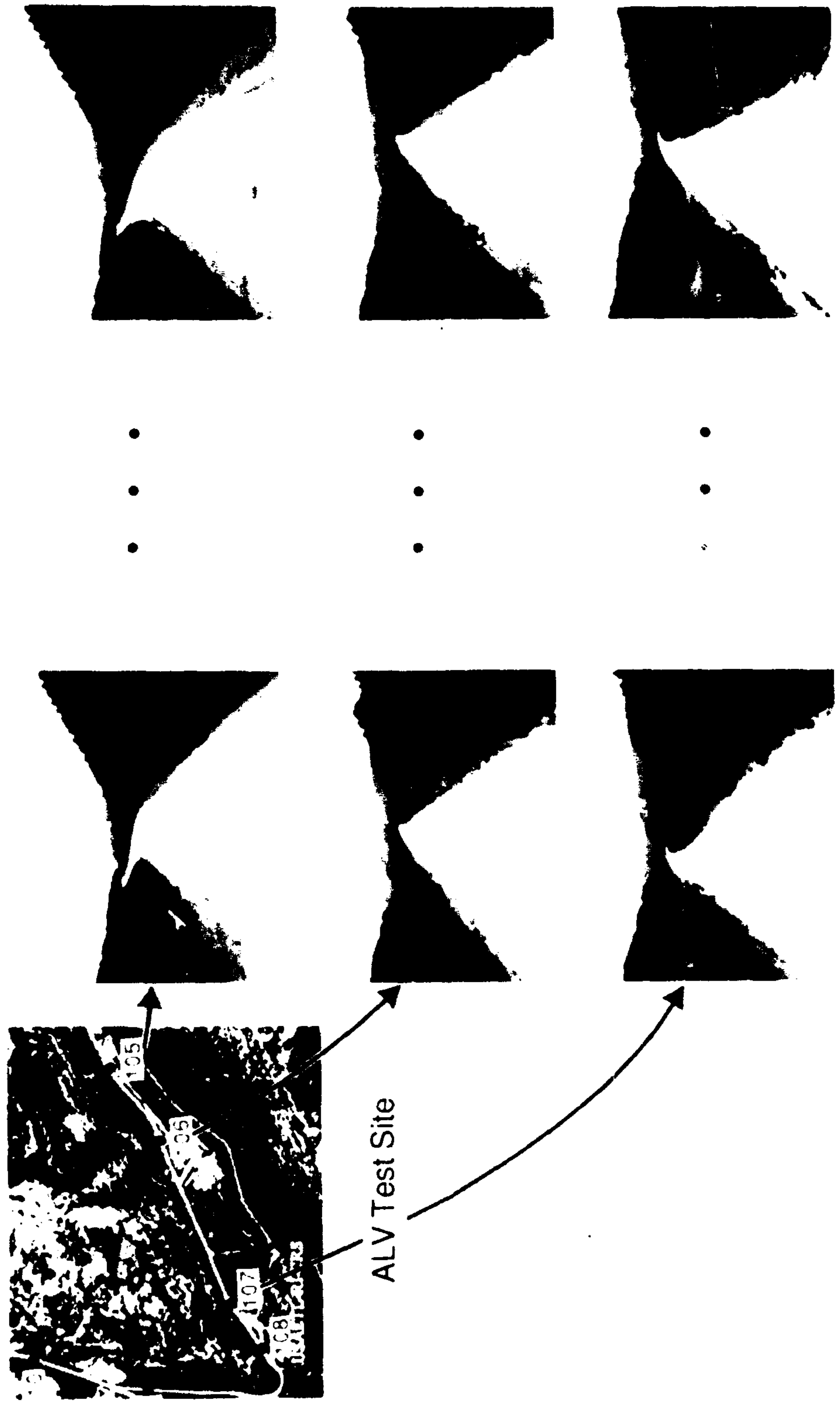
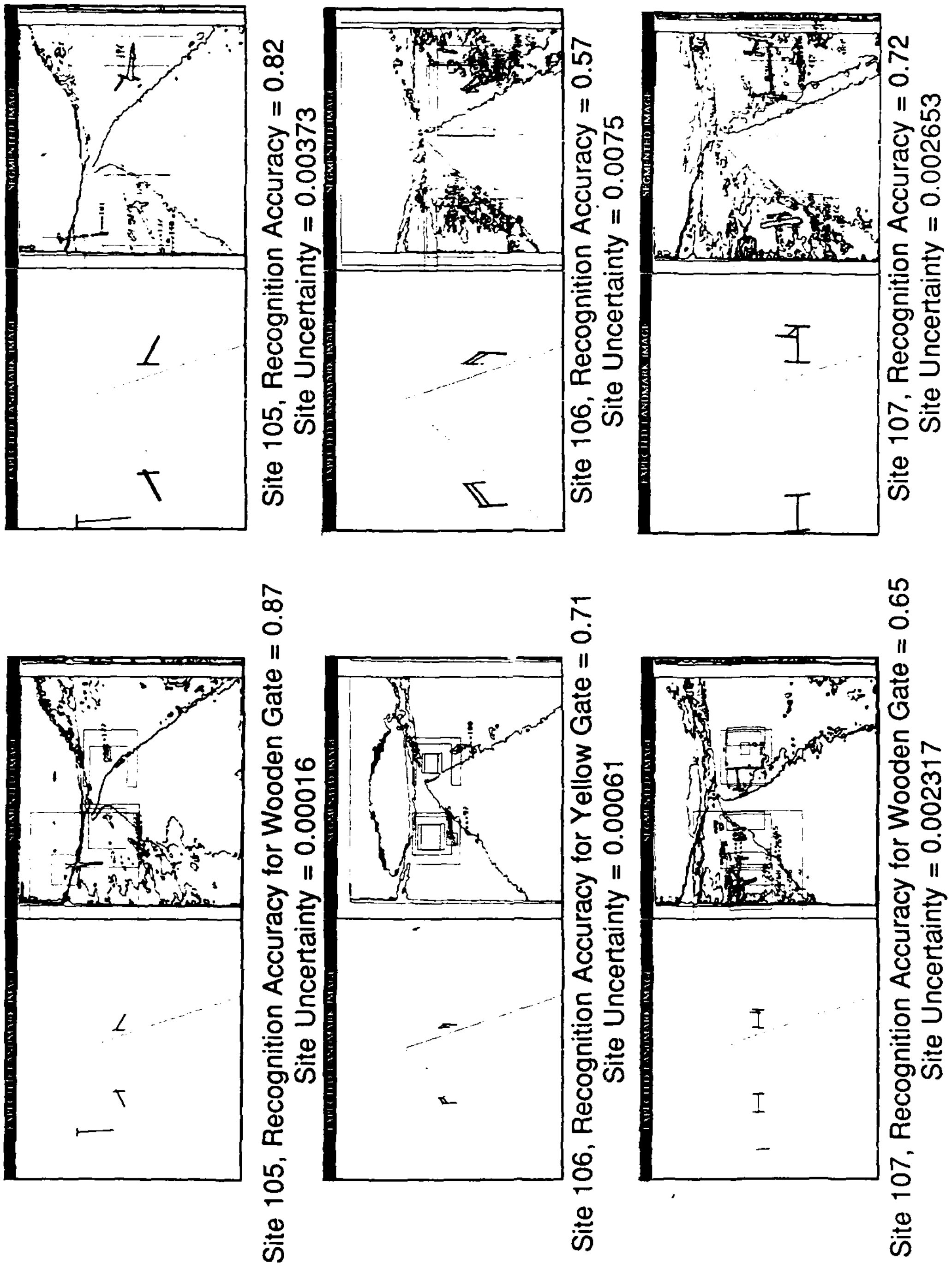


Figure 5: Original image sequences from three ALV test sites.



Site 105, Recognition Accuracy for Wooden Gate = 0.87
Site Uncertainty = 0.00016

Site 106, Recognition Accuracy for Yellow Gate = 0.71
Site Uncertainty = 0.00061

Site 107, Recognition Accuracy for Wooden Gate = 0.65
Site Uncertainty = 0.002317

Site 105, Recognition Accuracy = 0.82
Site Uncertainty = 0.00373

Site 106, Recognition Accuracy = 0.57
Site Uncertainty = 0.0075

Site 107, Recognition Accuracy = 0.72
Site Uncertainty = 0.002653

Figure 6: Dynamic model matching results for the three sites shown in Figure 5. Recognition accuracy for each landmark and the resulting site uncertainty are indicated.

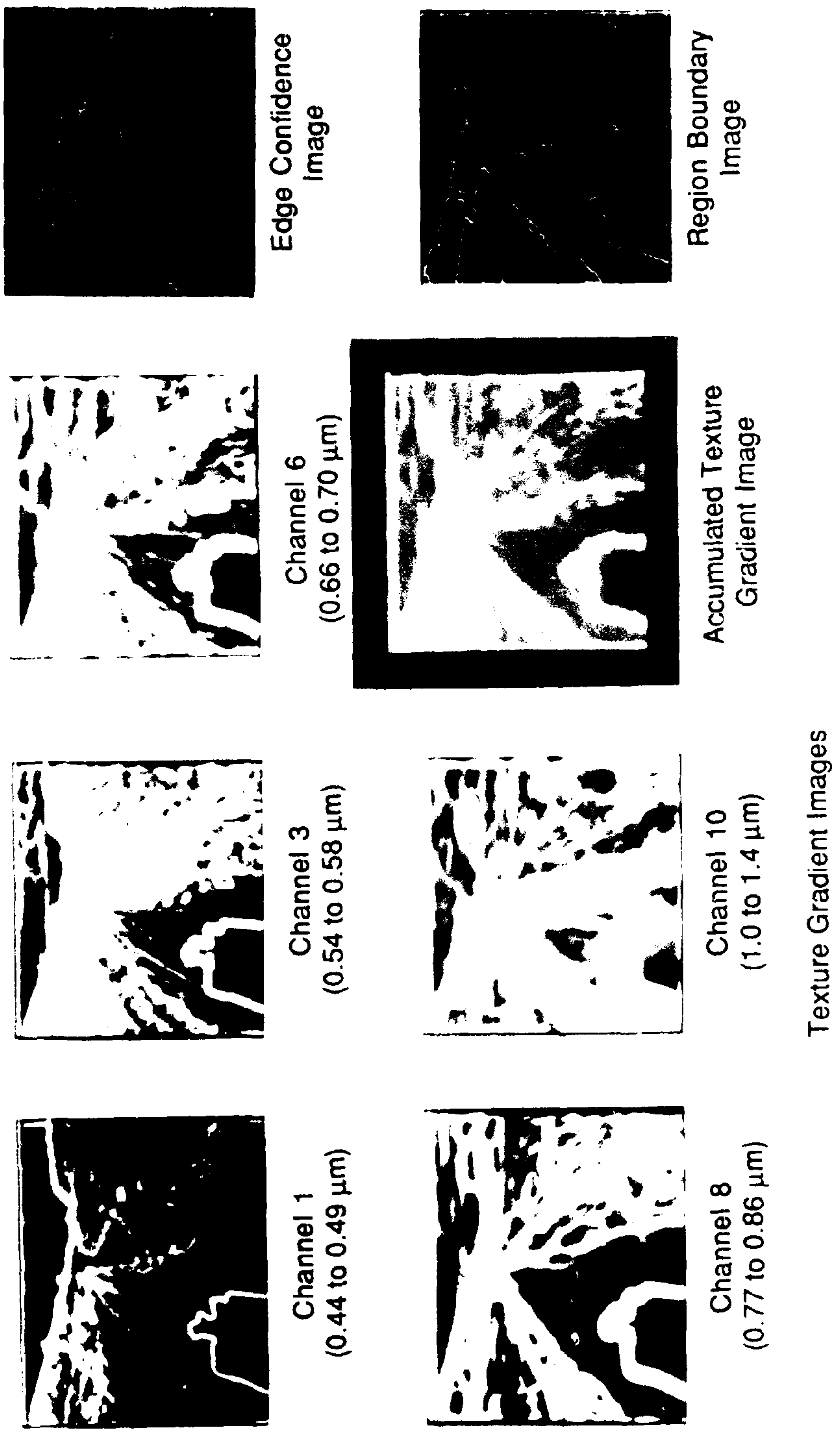


Figure 7: Texture gradient images anrl region boundaries using multispectral imagery

automatically acquire new target models and update their descriptions; to learn new target features based on perceptual cues; and to adapt segmentation parameters using genetic algorithms.[7]

Through an in-depth analysis, performed by Honeywell[2] on the applicability of state-of-the-art machine learning technology to model-based vision, we have developed the concepts for a novel machine learning system, called ORACLE (Object Recognition Accomplished through Consolidated Learning Expertise). The ORACLE system incorporates *explanation-based learning, structured conceptual clustering, genetic algorithms*, and *learning by examples* into a multi-strategy learning approach for automated target recognition. At the high level of computer vision, ORACLE utilizes the characterization and aggregation capabilities of explanation-based learning, structured conceptual clustering, and similarity-based learning in the target recognition and learning process. By combining these three learning systems, ORACLE overcomes the inherent limitations of the individual techniques and provides solutions to practical problems in model-based vision technology such as the need for automated model acquisition and refinement. During the intermediate level vision processing, ORACLE uses explanation-based learning with a perceptual cue database to acquire new target features. Finally, at the low level of vision, ORACLE uses genetic algorithms for parameter adaptation capability. Thus, the ORACLE system provides a learning capability at all the three levels of vision: low, intermediate and high.

9. SCIENTIFIC PERFORMANCE EVALUATION OF VISION ALGORITHMS AND SYSTEM

At present, very little work has been performed in the area of evaluation for image understanding algorithms and systems. In the DARPA-sponsored image understanding research, a wide variety of algorithms and systems are being developed for photointerpretation, navigation, manufacturing, cartography, and targeting applications. Scientific (both quantitative and qualitative) performance evaluation of diverse vision algorithms and systems will help in advancing the computer vision field at a faster pace, which in turn will lead to the most rapid fielding of computer vision technology. Effective performance evaluation will allow the measurement of not only the qualitative advancements in the computer vision field, but will also help to quantify the progress in the field. Most importantly, scientific performance evaluation will provide more rapid technology transfer (see Figure 8) to DoD applications by reducing the time needed to develop and validate robust vision algorithms. Figure 9 and 10 show algorithm evolution cycle and the evolution of algorithms. The four phases of the evolution cycle are conceptualization, generation, evaluation and adaptation. Once the algorithms have shown their *potential* value, they can be subjected to automatic evaluation.

Life cycle of any technology consists of four phases as shown in Figure 11. The maturity of any area (application areas or low, intermediate or high levels) of image understanding is related to the *degree* to which agreement on benchmarks can be reached and performance evaluation can be conducted.

Objective of performance evaluation is not to find out "the best" algorithm, but quantitative/qualitative understanding of capabilities of algorithms and systems. Evaluation works best when it is not tailored for a particular implementation; time to run the test is short; no new systems are designed; it has extreme cases especially those that cause known algorithms to fail; it has as extensive set of test images; anyone can submit results; and researchers perform the test on their own system.

It is extremely important to develop quantitative performance criteria for image understanding algorithms for several reasons:

(1) To compare various "matured" algorithms and systems and to predict their performance in a given scenario and/or for a specific application,

(2) To study the behavior of an application system and its components under different conditions and parameter settings, so as to be able to find the optimum performance achievable and the performance bounds of its components,

(3) To understand the characteristics of the imagery that affect the performance of the algorithms,

(4) To find common functional elements for an application among the algorithms currently in use,

(5) To help the algorithm developer choose the appropriate algorithms for his/her application and research, and

(6) To provide an objective and complete evaluation methodology for standardization purposes.

Performance evaluation allows performance analysis (strengths/weaknesses), sensitivity analysis, performance models. All these lead to prediction of algorithms and prediction is an important element of science.

The critical ingredients for scientific performance evaluation are:

(a) Image database groundtruth,

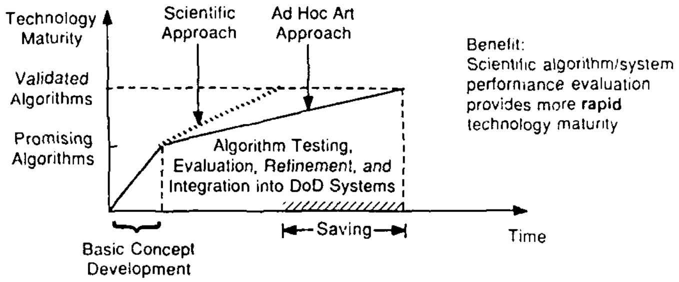


Figure 8: Scientific approach to technology evaluation.

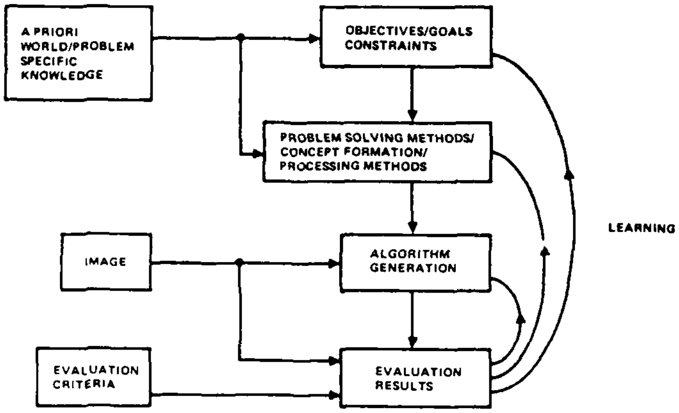

A PRIORI WORLD/PROBLEM SPECIFIC KNOWLEDGE
OBJECTIVES/GOALS CONSTRAINTS
PROBLEM SOLVING METHODS/ CONCEPT FORMATION/ PROCESSING METHODS
IMAGE
ALGORITHM GENERATION
EVALUATION CRITERIA
EVALUATION RESULTS
LEARNING

Figure 9: Algorithm evolution cycle.

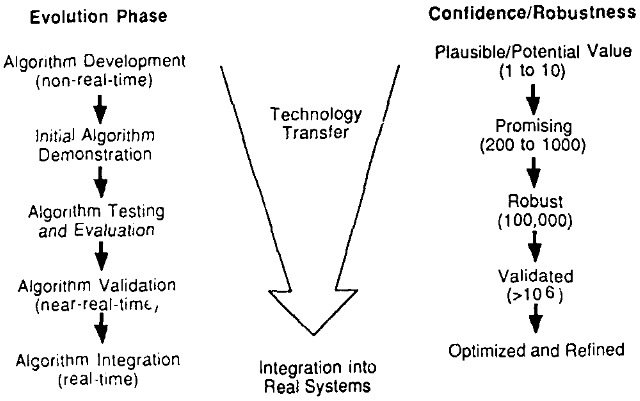

Evolution Phase
Algorithm Development (non-real-time)
Initial Algorithm Demonstration
Algorithm Testing and Evaluation
Algorithm Validation (near-real-time)
Algorithm Integration (real-time)
Technology Transfer
Integration into Real Systems
Confidence/Robustness
Plausible/Potential Value (1 to 10)
Promising (200 to 1000)
Robust (100,000)
Validated (>10^6)
Optimized and Refined

Figure 10: Evolution of algorithms.

Basic Research →	Applied Research →	Development Phase →	Engineering/Phase
Concept Formation	Characterized by benchmarks & evaluation	Uses benchmarks for developing an application	Operational System

Figure 11: Life cycle of technology development.

(b) Techniques for performance evaluation,

(c) Common system environments (KBVision and others).

Some of the problems with performance evaluation are:

1. Lack of appropriate database,
2. Slightly different problem statements and assumptions need different data sets,
3. Difficulty to quantify algorithm performance, many facets of the problem,
4. Interpretation of evaluation results (who?)

There are two solutions: natural evolution or concrete action to promote the maturation of IU tech base. We concentrate on the second solution.

We firmly believe that the effective characterization and prediction of algorithm performance is an essential step in transforming computer vision from an art to a science. The ability to successfully predict performance depends on a clear understanding of the complex relationship among the input data, algorithm operations, and produced results.

Through active interaction with the DARPA IU/SC community, the following objectives are pursued for scientific performance evaluation.

(1) National research database of computer vision imagery,

(2) Characterization (taxonomy) of vision research,

(3) Benchmarks for performance measurement of algorithms/systems (what to measure?),

(4) Techniques and models for "matured" algorithms and systems for performance evaluation (how to measure?),

(5) Workshop of DARPA IU community on performance evaluation.

The details of the above objectives are now discussed.

9.1 NATIONAL RESEARCH DATABASE OF COMPUTER VISION IMAGERY

The objective of establishing a national research image database is to promote the orderly development and dissemination of image information to serve the needs of DARPA IU algorithms/systems developers. This encompasses the standards for data interchange and activities for data collection, data organization, and data distribution.

The important considerations for these databases are: ground truth data requirements (site, sensor characterization, sensor platform, objects of interest, meteorological conditions), ground truth recording procedures, and database quantity/quality/variety requirements. The ground truth information is very critical and many times is not available or is too expensive to capture. Whenever the ground truth information is available, imagery should be partitioned into two categories: For some imagery, the ground truth is supplied to the researcher so that he/she can use them in the development of vision algorithms; the other category should be the imagery for which ground truth is sequestered and used to evaluate the robustness of the algorithms after development.

One potential use of an accepted imagery database would be for evaluating various "matured" algorithms that perform the same function (e.g., stereo, segmentation, motion detection, object recognition in range images, etc). Each year, the results of this evaluation, which have a well defined objective and scientific experiment, can be publicized at the Image Understanding Workshop and there would be recognition for researchers who demonstrate the best results using the "matured" algorithms for a given set of images (for a given application). More importantly, the overall progress made by the IU community would be made apparent.

Our short-term objective is to define and make available a standard set of images to be used by the DARPA IU community for algorithm development and evaluation. Some of the common functions are: stereo, motion algorithms, object recognition in outdoor/indoor scenes using TV and range images, etc.

Some of the currently available databases that may meet our needs are:

Martin's Collage 1 and Collage 2 ALV Database: Contains a large number of color TV, multispectral, and range images. Ground truth information is limited.

USC Database: Contains a large number of texture, aerial, and color images. Ground truth is limited.

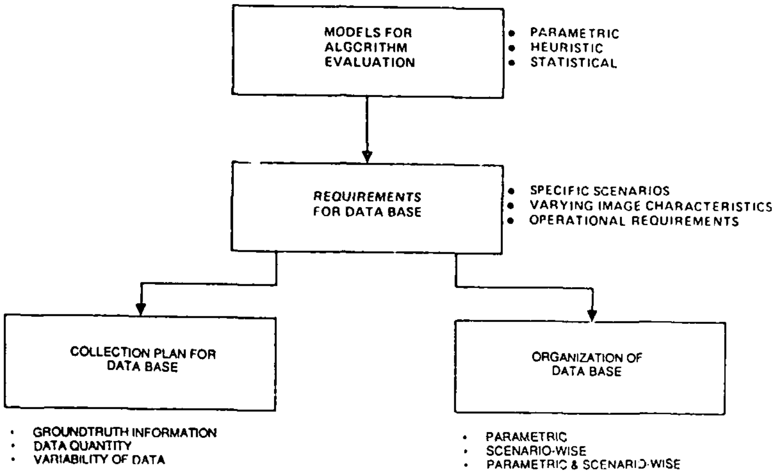

MODELS FOR ALGORITHM EVALUATION
• PARAMETRIC
• HEURISTIC
• STATISTICAL
REQUIREMENTS FOR DATA BASE
• SPECIFIC SCENARIOS
• VARYING IMAGE CHARACTERISTICS
• OPERATIONAL REQUIREMENTS
COLLECTION PLAN FOR DATA BASE
• GROUNDTRUTH INFORMATION
• DATA QUANTITY
• VARIABILITY OF DATA
ORGANIZATION OF DATA BASE
• PARAMETRIC
• SCENARIO-WISE
• PARAMETRIC & SCENARIO-WISE

Figure 12: Interaction of algorithms and database.

CMU: NavLab, Calibrated Imaging Lab

Martin Marietta: ALV database

Utah Range Database: Contains four sets of 33 images (5.108 Megabytes). Data is available in Unix TAR format, 1600 BPI tape. This data includes:

> *University of Utah Images:* The set is encoded in White Scanner format. Consists of images of bottles, cylinders, polygons, and various parts including the Renault auto part from INRIA. The parameters and details of the scanning systems are known for the set of images scanned at Utah.
>
> *SRI Images:* Grapes and space shuttle images.
>
> *North Carolina State University:* PC board, the image of Victor Hugo, and others.
>
> *ENST Paris:* Victor Hugo image obtained in one degree aspect increments.

The sample image sets can be separated by class as well. In the case of stereo imagery, the photogrammetry community has distributed a very good stereo database (not epi-polar constrained) with known ground truth. Imagery acceptable for motion research is available from SRI, which has some image sequences that have good imaging information and some partial depth ground truth.

Note that database is closely tied up with evaluation (see Figure 12) Database should also be viewed as extensible, not the finished product. In summary some of the issues related with database are:

(1) Models for evaluation,

(2) Requirements of database

(3) Collection of database,

(4) Organization of database,

(5) Maintenance of database,

(6) Access and usage expectations,

(7) Groundtruth information - sensor, map, other ancillary information,

(8) Standards for imagery and non-imagery information,

(9) Types of data,

(10) Specific IU algorithms, systems and applications.

We are working on a detailed plan for data acquisition and accessibility. We are identifying core data set and plan to expand it in a systematic way.

9.2 Characterization of Vision Research Infrastructure or Taxonomies of Vision Research

A current detailed taxonomy of vision research is desired which is based on diverse criteria such as:

- Applications (navigation, terminal homing, remote sensing, etc.),
- Class of sensors (TV, FLIR, Range, SAR, etc.),
- Use of multisensors (TV & range, FLIR & Range, etc.),
- Functionalities (segmentation, feature extraction, texture analysis/synthesis, etc.),
- Principles (top down, bottom up, etc.),
- Reasoning processes (qualitative, perceptual, etc.),
- Hardware architectures (systolic, array, cellular, etc.),
- Implementation techniques (VHSIC, VLSI, discrete, etc.), and
- Use of auxiliary information (digital elevation map data, land cover data, etc.).

The rationale for characterization is to help in the organization and development of image database, definition of benchmarks and methodologies for evaluation. This characterization will provide a common framework of terminology and description to promote improved communication among the members of the vision community and between technology developers and appliers. Since the computer vision field is still quite young and undergoing rapid evolution, the proposed taxonomy should be viewed as a "snapshot" of the field today and will likely need to undergo significant modifications and extensions as the field progresses. After the development of the proposed taxonomy, the development of the other three related goals will be pursued: a common image database, general vision system benchmarks, and an effective methodology for performance evaluation. One can think of a very deep tree whose leaf nodes are very specific (for example, the segmentation of tank targets at "close" distances in range images for terminal homing applications). We associate the specific database, benchmarks, and methodology with these leaf nodes for performance evaluation.

9.3 Terminology and Benchmarks for Performance Evaluation

It is important to have common terminology and benchmarks for performance evaluation. Subtle differences in meaning can be very important for evaluation. Even terms such as "ground truth" mean different things to different people. A lexicon that establishes standard terminology and standard benchmarks will provide uniformity in carrying out scientific experiments for performance evaluation.

9.4 Scientific Methodology and Models for Performance Evaluation

Since one of the goals of computer vision is to build machines that can solve real world problems, we need to define the *systematic methods and models for performance* evaluation of individual vision algorithms (segmentation, feature computation, texture measurement, etc.) and systems (object recognition, vision-based navigation, etc.) for a particular application (terminal homing, surveillance, etc.). We need to thoroughly understand the practical experimental designs and the errors of observation and their treatment. As an example, the results of segmentation are still evaluated qualitatively. They need to be evaluated, in part, on the basis of how well the implicit or explicit model in the technique is able to predict performance. In other words, the quality of an algorithm should depend not only on certain test performance results, but on the accuracy of the model that predicts algorithm performance over a diverse database of imagery. If an algorithm performs well over a narrow database (a few images), but a resulting performance prediction model proves to be inaccurate over va larger database, the proper conclusion is that the overall algorithm performance is deficient. In this framework, evaluation of an algorithm consists of two components: an algorithm and the associated performance prediction model. We can refer to the combination of these two components as a *generalized algorithm*. Simple quantitative measures (which may or may not have intuitive physical significance), such as the number of pixels misclassified with respect to the true object, the correlation coefficient between the true and extracted object, mean square error between the true and extracted object, object-to-background contrast, and the metric based on these criteria, can be effectively used for segmentation evaluation.

Carrying this evaluation process a step further, we need an evaluation methodology for the evaluation of complete systems such as an object recognition system. The system performance should be evaluated on the basis of the task it is able to perform in a given environment considering such factors as sensor type, resolution of data, type of objects, and complexity and information content of the scene. It is logical to assume that to obtain the optimum performance of the system, it is essential to achieve the maximum attainable performance of each of its components. However, it may or may not satisfy the goals of the system performance, since most of the image understanding components are nonlinear. Here a top-down approach for evaluation may be more meaningful than a bottom-up approach.

In summary, the emphasis of performance evaluation is on *computer* vision problems, scientific *experimental design* and *interfaces* between vision components and functions. We need to define a performance metric for each of the image understanding algorithms as well as a performance metric for the system as a whole. This can be done for the specific matured algorithms/systems being pursued by the Image Understanding community.

ACKNOWLEDGEMENTS

The work summarized in this report is supported by DARPA under contract DACA 76-86-C-0017, by the NASA under contract NAS2-12800, and other sponsors. I would like to thank Bob Simpson, Takeo Kanade, Ram Nevatia, Keith Price, Ed Riseman Tom Binford, Steve Shafer, Chuck Thorpe, Karen Olin, Bob Bolles, and Steve Savitt for helpful comments and suggestions on Scientific Performance Evaluation.

The following researchers have contributed to the work described in this report: P. Symosek, H. Nasr, W. Burger, J. Ming, B. Roberts, J. Kim, H. Lai, S. Schaffer, and D. Robohm.

REFERENCES

1. B. Bhanu, "Automatic Target Recognition: State of the Art Survey," *IEEE Transactions on Aerospace & Electronic Systems* **AES-22**(4) pp. 364-379 (July 1986).

2. B. Bhanu, "Machine Learning in Computer Vision," Technical Report, Honeywell Systems and Research Center (1988).

3. B. Bhanu and W. Burger, "Qualitative Reasoning in Dynamic Scene Understanding," *Submitted to Computer Vision, Graphics and Image Processing*, (1987).

4. B. Bhanu and W. Burger, "Qualitative Motion Detection and Tracking of Targets from a Mobile Platform," *Proc. DARPA Image Understanding Workshop*, pp. 289-318 (April 1988).

5. B. Bhanu and W. Burger, "A System for Computing the Self Motion of Moving Imaging Devices," *Patent pending*, (1988).

6. B. Bhanu and W. Burger, "A System for Motion Detection and Tracking of Targets from a Mobile Platform," *Patent pending*, (1988).

7. B. Bhanu, S. Lee, and J. Ming, "Adaptive Image Segmentation Using A Genetic Algorithm," *Proc. DARPA Image Understanding Workshop*, Morgan Kaufmann Publishers, (May 1989).

8. B. Bhanu and J.C. Ming, "TRIPLE: A Multi-Strategy Machine Learning Approach to Target Recognition," Proc. DARPA Image Understanding Workshop, pp. 537-547 (April 1988).

9. B. Bhanu and B. Roberts, "Obstacle Detection During Rotorcraft Low Altitude Flight," Annual Technical Report for NASA-Ames (April 1989).

10. B. Bhanu, B. Roberts, and J. Ming, "Inertial Navigation Sensor Integrated Motion Analysis," *Proc. DARPA Image Understanding Workshop*, Morgan Kaufmann Publishers, (May 1989).

11. B. Bhanu and P. Symosek, "Interpretation of Terrain Using Hierarchical Symbolic Grouping fr [illegible]-Spectral Images," *Proc. DARPA Image Understanding Workshop*, pp. 466-474 (Feb. 1987).

12. B. Bhanu and P. Symosek, "Interpretation of Terrain Using Multispectral Images," *Submitted to Pattern Recognition*, (1989).

13. B. Bhanu, P. Symosek, J. Ming, W. Burger, H. Nasr, and J. Kim, "Qualitative Target Motion Detection and Tracking," *Proc. DARPA Image Understanding Workshop*, Morgan Kaufmann Publishers, (May 1989).

14. W. Burger and B. Bhanu, "Qualitative Motion Understanding," *Proc. Tenth International Joint Conference on Artificial Intelligence, IJCAI-87, Milan, Italy*, Morgan Kaufmann Publishers, (August 1987).

15. W. Burger and B. Bhanu, "Dynamic Scene Understanding for Autonomous Mobile Robots," *Proc. IEEE Conference on Computer Vision and Pattern Recognition*, (June 1988).

16. W. Burger and B. Bhanu, "Estimating 3-D Egomotion from Perspective Image Sequences," *Submitted to IEEE Transactions on Pattern Analysis and Machine Intelligence*, (1988).

17. W. Burger and B. Bhanu, "On Computing a 'Fuzzy' Focus of Expansion for Autonomous Navigation," *Proc. IEEE Conference on Computer Vision and Pattern Recognition*, (June 1989).

18. W. Burger and B. Bhanu, "Qualitative Understanding of Scene Dynamics for Autonomous Mobile Robots," *Submitted to International Journal of Robotics Research*, (1989).

19. R.M. Haralick and L.G. Shapiro, "Image Segmentation Techniques," *Computer Vision, Graphics and Image Processing* **29** pp. 100-132 (1985).

20. H. Nasr and B. Bhanu, "Dynamic Model Matching for Target Recognition from a Mobile Platform," *Proc. DARPA Image Understanding Workshop*, pp. 527-536 (April 1988).

21. H. Nasr and B. Bhanu, "Landmark Recognition System Using Dynamic Model Matching," *Patent pending*, (1988).

SECTION II

SPECIAL TALKS

AND

PRESENTED TECHNICAL

PAPERS

A Report on the Results of the DARPA Integrated Image Understanding Benchmark Exercise*

Charles Weems, Edward Riseman, Allen Hanson
Computer and Information Science Department
University of Massachusetts at Amherst

Azriel Rosenfeld
Center for Automation Research
University of Maryland

Abstract

This paper describes the second in a series of DARPA-sponsored efforts to evaluate the merits of various parallel architectures as applied to the problem of knowledge-based machine vision. The first DARPA benchmark considered only the execution times for a set of isolated vision-related tasks. However, the overall performance of an image interpretation system depends upon more than the execution times of a few key tasks. In particular, the costs of interactions between tasks, input and output, and system overhead must be taken into consideration. Thus, this new benchmark addresses the issue of system performance on an integrated set of tasks, where the task interactions are typical of those found in complex vision applications. Also, unlike the majority of benchmarks, which are numerically oriented and whose components are small, stand-alone programs, this benchmark tests system performance on many types of processing and in the context of a larger program. As a result, the benchmark can be used to gain insight into a greater variety of processor capabilities and foibles, and may thus help to guide the development of the next generation of parallel vision architectures.

Introduction

Knowledge-based image understanding presents an immense computational challenge that has yet to be satisfied by any parallel architecture. The challenge is not merely to provide a greater quantity of operations per second, but also to supply the necessary varieties of operations in the required amounts. Consider that a sequence of images at medium resolution (512 x 512 pixels) and standard frame rate (30 frames per second) in color (24 bits per pixel) represents a data input rate of about 23.6 million bytes per second and, in a typical interpretation scenario, many thousands of operations may be applied to each input pixel in order to enhance, and segment an image and to extract various features from it. But in addition to these I/O and pixel processing requirements, a vision system must be able to do much more. For example, it must organize extracted image features via perceptual grouping mechanisms, locate relevant models in a potentially vast store of knowledge and compare them to partial models derived from the input data, generate hypotheses concerning the environment of the sensor, resolve conflicting hypotheses to arrive at a consistent interpretation of the environment, manage and update stored knowledge, and so on.

While traditional supercomputing benchmarks may be useful in estimating the performance of an architecture on some types of image processing tasks, those berchmarks have little relevance to the majority of the processing that takes place in a vision system [Duff, 1986]. Nor has there been much effort to define a vision benchmark for supercomputers, since those machines in their traditional form have usually been viewed as inappropriate vehicles for knowledge-based vision research. However, now that parallel processors are becoming readily available, and because they are viewed as being better suited to vision processing, researchers in both machine vision and parallel architecture are

* This work was supported in part by the Defense Advanced Research Projects Agency under contract number DACA76-86-C-0015, monitored by the U.S. Army Engineer Topographic Laboratories.

taking an interest in performance issues with respect to vision. The next section summarizes the work that has been done in the area of vision benchmarks to date.

Review of Previous Vision Benchmark Efforts

One of the first parallel processor benchmarks to address vision-related processing was the Abingdon Cross benchmark, defined at the 1982 Multicomputer Workshop in Abingdon, England [Preston, 1986]. In that benchmark, an input image was specified that consisted of a dark background with a pair of brighter rectangular bars, equal in size, that cross at their midpoints and are centered in the image, and with Gaussian noise added to the entire image. The goal of the exercise was to determine and draw the medial axis of the cross formed by the two bars. The results obtained from solving the benchmark problem on various machines were presented at the 1984 Multicomputer Workshop in Tanque Verde, Arizona, and many of the participants spent a fairly lengthy session discussing problems with the benchmark and designing a new benchmark that it was hoped would solve those problems.

It was the perception of the Tanque Verde group that the major drawback of the Abingdon Cross was its lack of breadth. The problem required a reasonably small repertoire of image processing operations to construct a solution. The second concern of the group was that the specification did not constrain the a priori information that could be used to solve the problem. In theory, a valid solution would have been to simply draw the medial lines since their true positions were known. Although this was never done, there was argument over whether it was acceptable for a solution to make use of the fact that the bars were oriented horizontally and vertically in the image. A final concern was that no method was prescribed for solving the problem, with the result that every solution was based on a different method. When a benchmark can be solved in different ways, the performance measurements become more difficult to compare because they include an element of programmer cleverness. Also, the use of a consistent method would permit some comparison of the basic operations that make up a complete solution.

The Tanque Verde group specified a new benchmark, called the Tanque Verde Suite, that consisted of a large collection of individual vision-related problems. Table 1 contains the list of problems that was developed. Each of the problems was to be further defined by a member of the group, who would also generate test data for their assigned problem. Unfortunately, only a few of the problems were ever developed, and none of them were widely tested on different architectures. Thus, while the simplicity of the Abingdon Cross may have been criticized, it was the respondent complexity of the Tanque Verde Suite that inhibited the latter's use.

Standard Utilities	High Level Tasks
3x3 Separable Convolution	Edge Finding
3x3 General Convolution	Line Finding
15x15 Separable Convolution	Corner Finding
15x15 General Convolution	Noise Removal
Affine Transform	Generalized Abingdon Cross
Discrete Fourier Transform	Segmentation
3x3 Median Filter	Line Parameter Extraction
256 Bin Histogram	Deblurring
Subtract Two Images	Classification
Arctangent(Image1/Image2)	Printed Circuit Inspection
Hough Transform	Stereo Image Matching
Euclidean Distance Transform	Camera Motion Estimation
	Shape Identification

Table 1: Tanque Verde Benchmark Suite

In 1986, a new benchmark was developed at the request of the Defense Advanced Research Projects Agency (DARPA). Like the Tanque Verde Suite, it was a collection of vision-related problems, but the set of problems that made up the new benchmark was much smaller and easier to implement. Table 2 lists the problems that comprised the first DARPA Image Understanding Benchmark. A workshop was held in Washington, D.C., in November of 1986 to present the results of testing the benchmark on several machines, with those results summarized in [Rosenfeld, 1987]. The consensus of the workshop participants was that the results cannot be compared directly for several reasons. First, as with the Abingdon Cross, no method was specified for solving any of the problems. Thus, in many cases, the timings were more indicative of the knowledge or cleverness of the programmer, than of a machine's true capabilities. Second, no input data was provided and the specifications allowed a wide range of possible inputs. Thus, some participants chose to test a worst-case input, while others chose "average" input values that varied considerably in difficulty.

11x11 Gaussian Convolution of a 512x512 8-bit Image
Detection of Zero Crossings in a Difference of Gaussians Image
Construct and Output Border Pixel List
Label Connected Components in a Binary Image
Hough Transform of a Binary Image
Convex Hull of 1000 Points in 2-D Real Space
Voronoi Diagram of 1000 Points in 2-D Real Space
Minimal Spanning Tree Across 1000 Points in 2-D Real Space
Visibility of Vertices for 1000 Triangles in 3-D Real Space
Minimum Cost Path Through a Weighted Graph of 1000 Nodes of Order 100
Find all Isomorphisms of a 100 Node Graph in a 1000 Node Graph

Table 2: Tasks from the First DARPA Image Understanding Benchmark

The workshop participants pointed out other shortcomings of the benchmark. Chief among these was that because it consisted of isolated tasks, the benchmark did not measure performance related to the interactions between the components of a vision system. For example, there might be a particularly fast solution to a problem on a given architecture if the input data is arranged in a special manner. However, this apparent advantage might be inconsequential if a vision system does not normally use the data in such an arrangement, and the cost of rearranging the data is high. Another shortcoming was that the problems had not been solved before they were distributed. Thus, there was no canonical solution on which the participants could rely for a definition of correctness, and there was even one problem for which it turned out there was no practical solution. The issue of having a ground truth, or known correct solution was considered very important, since it is difficult to compare the performance of two architectures when they produce different results. For example, is an architecture that performs a task in half the time of another really twice as powerful if the first machine's programmer used integer arithmetic while the second machine was programmed to use floating point, and they thus obtained significantly different results? Since problems in vision are often ill-defined, it is possible to argue for the correctness of many different solutions. In a benchmark, however, the goal is not to solve a vision problem but to test the performance of different machines doing comparable work.

The conclusions from the first DARPA benchmark exercise were that the results should not be directly compared, and that a new benchmark should be developed that addresses the shortcomings of the preceding benchmarks. Specifically, the new benchmark should test system performance on a task that approximates an integrated solution to a machine vision problem. A complete solution with test data sets should be constructed and distributed with the benchmark specification. And, every effort should be made to specify the the benchmark in such a way as to minimize the opportunities for taking shortcuts in solving the problem. The task of constructing the new benchmark, to be called the Integrated Image Understanding Benchmark, was assigned to the vision research groups at the University of Massachusetts at Amherst, and the University of Maryland.

Following the 1986 meeting, a preliminary benchmark specification was drawn up and circulated among the DARPA image understanding community for comment. The benchmark specification was then revised, and a solution was programmed on a standard sequential machine. In creating the solution, several problems were discovered and the benchmark specification was modified to correct those problems. The programming of the solution was done by the group at the University of Massachusetts and the code was then sent to the group at the University of Maryland to verify its validity, portability, and quality. The group at Maryland also reviewed the solution to verify that it was general in nature and neutral with respect to any underlying architectural assumptions. The Massachusetts group developed a set of five test cases, and a sample parallel solution for a commercial multiprocessor.

In March of 1988, the benchmark was released, and made available from Maryland via network access, or by sending a blank tape to the group in Massachusetts. The benchmark release consisted of the sequential and parallel solutions, the five test cases, and software for generating additional test data. The benchmark specification was presented at the DARPA Image Understanding Workshop, the International Supercomputing Conference, and the Computer Vision and Pattern Recognition conference [Weems, 1988]. Over 25 academic and industrial groups, listed in Table 3, obtained copies of the benchmark release. Nine of those groups developed either complete or partial versions of the solution for an architecture. A workshop was held in October of 1988, in Avon Old Farms, Connecticut, to present those results to members of the DARPA research community. As with the previous workshops, the participants spent a session developing a critique of the benchmark and making recommendations for the design of the next version.

International Parallel Machines	Hughes AI Center
Mercury Computer Systems	University of Wisconsin
Stellar Computer	George Washington University
Myrias Computer	University of Massachusetts*
Active Memory Technology	SAIC
Thinking Machines*	Eastman Kodak
Aspex Ltd.*	University College London
Texas Instruments	Encore Computer
IBM	MIT
Carnegie-Mellon University*	University of Rochester
Intel Scientific Computers*	University of Illinois*
Cray Research	University of Texas at Austin*
Sequent Computer Systems*	Alliant Computer*

Table 3: Distribution List for the Second DARPA Benchmark
*** Indicates Results Presented at the Avon Workshop**

The remainder of this paper summarizes those results and recommendations, following a brief review of the benchmark task and the rationale behind its design.

Benchmark Task Overview

The overall task that is to be performed by this benchmark is the recognition of an approximately specified 2 1/2 dimensional "mobile" sculpture in a cluttered environment, given images from intensity and range sensors. The intention of the benchmark designers is that neither of the input images, by itself, is sufficient to complete the task.

The sculpture to be recognized is a collection of two-dimensional rectangles of various sizes, brightnesses, two-dimensional orientations, and depths. Each rectangle is oriented normal to the Z axis (the viewing axis), with constant depth across its surface, and the images are constructed under orthographic projection. Thus an individual rectangle has no intrinsic depth component, but depth is a factor in the spatial relationships between rectangles. Hence the notion that the sculpture is 2 1/2 dimensional.

The clutter in the scene consists of additional rectangles, with sizes, brightnesses, two-dimensional orientations, and depths that are similar to those of the sculpture. Rectangles may partially or completely occlude other rectangles. It is also possible for a rectangle to disappear when another rectangle of the same brightness or slightly greater depth is located directly behind it.

A set of models is provided that represent a collection of similar sculptures, and the recognition task involves identifying the model which best matches the object present in the scene. The models are only approximate representations of sculptures in that they allow for slight variations in component rectangle's sizes, orientations, depths, and the spatial relationships between them. A model is constructed as a tree structure where the links in the tree represent the invisible links in the sculpture. Each node of the tree contains depth, size, orientation, and intensity information for a single rectangle. The child links of a node in the tree describe the spatial relationships between that node and certain other nodes below it.

The scenario that the designers imagined in constructing the problem was a semi-rigid "mobile", with invisible links, viewed from above, with bits and pieces of other mobiles blowing through the scene. The state of the system is that previous processing has narrowed the range of potential matches to a few similar sculptures, and has oriented them to correspond with information extracted from a previous image. However, the objects in the scene have since moved, and a new set of images has been taken prior to completing the matching process. The system must make its final choice for a best match, and update the corresponding model with the positional information extracted from the latest images.

The intensity and depth sensors are precisely registered with each other and both have a resolution of 512 x 512 pixels. There is no averaging or aliasing in either of the sensors. A pixel in the intensity image is an 8-bit integer grey value. In the depth image a pixel is a 32-bit floating-point range value. The intensity image is noise free, while the depth image has added Gaussian noise.

A set of test images is created by first selecting one of the models in a set. The model is then rotated and translated as a whole, and its individual elements are then perturbed slightly. Next, a collection of spurious rectangles is created with properties that are similar to those in the chosen model. All of the rectangles (both model and spurious) are then ordered by depth and drawn in the two image arrays. Lastly, an array of Gaussian-distribution noise is added to the depth image array.

Figure 1 shows an intensity image of a sculpture alone, and Figure 2 shows the sculpture with added clutter.

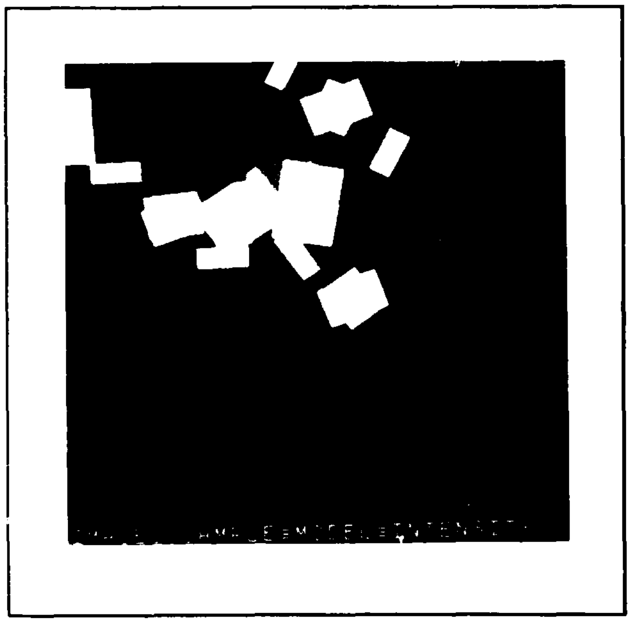


Figure 1: Intensity Image of Model Alone

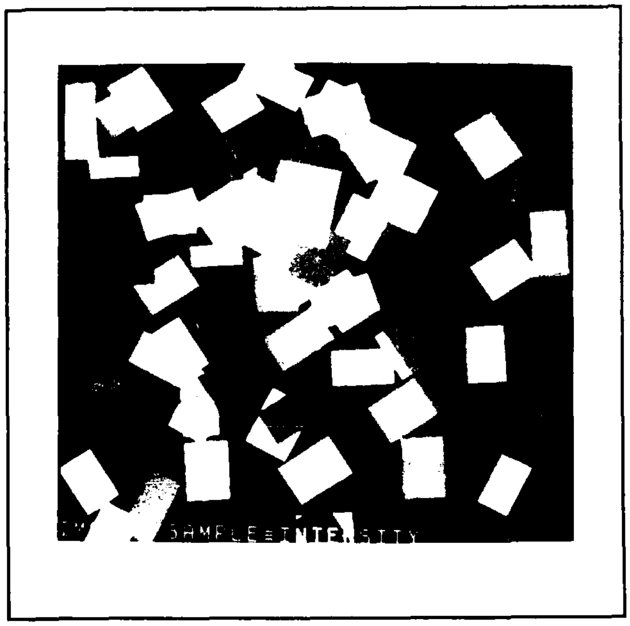


Figure 2: Image of Model with Clutter

Processing in the benchmark begins with some low-level operations on the intensity and depth images, followed by some grouping operations on the intensity data that result in the extraction of candidate rectangles. The candidate rectangles are used to form partial matches with the stored models. For each model, it is possible that multiple hypothetical poses will be established. The benchmark then proceeds through the model poses, using the stored information to probe the depth and intensity images in a top-down manner. Each probe can be thought of as testing an hypothesis for the existence of a rectangle in a given location in the images. Rejection of an hypothesis, which only occurs when there is strong evidence that a rectangle is actually absent, results in elimination of the corresponding model pose. Confirmation of the hypothesis results in the computation of a match strength for the rectangle at the hypothetical location, and an update of its representation in the model with new size, orientation, and position information. It is possible for the match strength to be as low as zero when there is no supporting evidence for the match and a lack of strong evidence that the rectangle is absent, as in the case of a rectangle that is entirely occluded by another. After a probe has been performed for every unmatched rectangle in the list of model poses, an average match strength is computed for each pose that has not been eliminated. The model pose with the highest average match strength is selected as the best match, and an image is generated that highlights the model in the intensity image. Table 4 lists all of the steps that make up the complete benchmark task.

The benchmark specification requires that this set of steps be applied in implementing a solution. Furthermore, for each step, a recommended method is described that should be followed whenever possible. However, in recognition of the fact that some methods simply may not work, or will be extremely inefficient for a given architecture, implementors are permitted to substitute other methods for individual steps. When it is necessary for an implementation to differ from the specification, the implementor is expected to supply a justification for the change. It is also urged that, if possible, a version of the implementation be written and tested with the recommended method so that the difference in performance can be determined.

Benchmark Philosophy and Rationale

In writing an integrated image understanding benchmark, the goal is to create an interpretation scenario that is an approximation of an actual image interpretation task. One must remember, however, that the benchmark problem is not an end in itself, but is a framework for testing machine performance on a wide variety of common vision operations and algorithms, both individually and in an integrated form that requires communication and control across algorithms and representations. This benchmark is not intended to be a challenging vision research exercise, and the designers feel that it should not be. Instead, it should be a balance between simplicity for the sake of implementation by participants, and the complexity that is representative of actual vision processing. At the same time, it must test machine performance in as many ways as possible. A further constraint on the design was the requirement that it make use of as many of the tasks from the first DARPA benchmark as possible, in order to take advantage of the previous programming effort.

The job of the designers was thus to balance these conflicting goals and constraints in developing the benchmark scenario. One result is that the benchmark solution is neither the most direct, nor the most efficient method of solving the problem. However, making the solution more direct would have eliminated several of the algorithms that are important in testing certain aspects of machine performance. One the other hand, increasing the complexity of the problem to necessitate the use of those algorithms would have required significant additional processing that is redundant in terms of performance evaluation. Thus, while the benchmark solution is not a good example of how to build an efficient vision system, it is an effective test of machine performance both on a wide variety of individual operations and on an integrated task. Moreover, having taken a lesson from the Tanque Verde Suite, the benchmark design attempts to minimize the effort required of the participants, while maximizing the information obtained.

Low-Level, Bottom-Up Processing	
Intensity Image	**Depth Image**
Label Connected Components	3x3 Median Filter
Compute K-Curvature	3x3 Sobel and Gradient Magnitude
Extract Corners	Threshold
Intermediate Level Processing	
Select Components with 3 or More Corners	
Convex Hull of Corners for Each Component	
Compute Angles Between Successive Corners on Convex Hulls	
Select Corners with K-Curvature and Computed Angles Indicating a Right Angle	
Label Components with 3 Contiguous Right Angles as Candidate Rectangles	
Compute Size, Orientation, Position, and Intensity for Each Candidate Rectangle	
Model-Based, Top-Down Processing	
Determine all Single Node Isomorphisms of Candidate Rectangles in Stored Models	
Create a List of all Potential Model Poses	
Perform a Match Strength Probe for all Single Node Isomorphisms (see below)	
Link Together all Single Node Isomorphisms	
Create a List of all Probes Required to Extend Each Partial Match	
Order the Probe List According to the Match Strength of the Partial Match Being Extended	
Perform a Probe of the Depth Data for Each Probe on the List (see below)	
Perform a Match Strength Probe for Each Confirming Depth Probe (see below)	
Update Rectangle Parameters in the Stored Model for Each Confirming Probe	
Propagate the Veto from a Rejecting Depth Probe Throughout the Corresponding Partial Match	
When No Probes Remain, Compute Average Match Strength for Each Remaining Model Pose	
Select Model with Highest Average Match Strength as the Best Match	
Create the Output Intensity Image, Showing the Matching Model	
Depth Probe	
Select an X-Y Oriented Window in the Depth Data that will Contain the Rectangle	
Perform a Hough Transform Within the Window	
Search the Hough Array for Strong Edges with the Approximate Expected Orientations	
If Fewer than 3 Edges are Found, Return the Original Model Data with a No-Match Flag	
If 3 Edges are Found, Infer the Fourth from the Model Data	
Compute New Size, Position, and Orientation Values for the Rectangle	
Match-Strength Probe	
Select an Oriented Window in the Depth Data that is Slightly Larger than the Rectangle	
Classify Depth Pixels as Too Close, Too Far, or In Range	
If the Number of Too Far Pixels Exceeds a Threshold, Return a Veto	
Otherwise, Select a Corresponding Window in the Intensity Image	
Select Intensity Pixels with the Correct Value	
Compute a Match Strength Based on the Number of Correct vs. Incorrect Pixels in the Images	

Table 4: Steps that Compose the Integrated Image Understanding Benchmark

The great variety of architectures to be tested is itself a complicating factor in the design of a benchmark. It was recognized that each architecture may have its own most efficient method for computing a given function. However, the purpose of the benchmark requires that the benchmark tasks and methods be well defined so that the results from different machines will be comparable. Otherwise the results will include a significant factor that depends on the cleverness of the programmer. Thus the benchmark specification requests that participants do not take shortcuts in the solution, and that they use the recommended methods whenever possible. It should be noted that the recommended methods are not always the most efficient techniques because they were chosen to be as widely implementable as possible. Thus, while the processing time for a given step or for the entire task may not be the best performance that a machine can muster, it will be comparable to the results from others. Participants were also encouraged to develop timings for more optimal solutions, in addition to the standard solution, if they so desired.

The designers also recognize the tendency for any benchmark to turn into a horse race. However, that is not the goal of this exercise, which is to increase the scientific insight of architects and vision researchers into the architectural requirements for knowledge-based image interpretation. To this end, the benchmark requires a much more extensive set of instrumentation than simple execution times. Participants are required to report execution time for individual tasks, for the entire task, for system overhead, input and output, system initialization and loading any precomputed data, and for different processor configurations if possible. Implementation factors that are to be reported include an estimate of the time spent implementing the benchmark, the number of lines of source code, the programming language used, and the size of the object code. Machine configuration and technology factors that are requested include the number of processors, memory capacity, data path widths, integration technology, clock and instruction rates, power consumption, physical size and weight, cost, and any limits to scaling-up the architecture. Lastly, participants are asked to comment on any changes to the architecture that they feel would contribute to an improvement in performance on the benchmark.

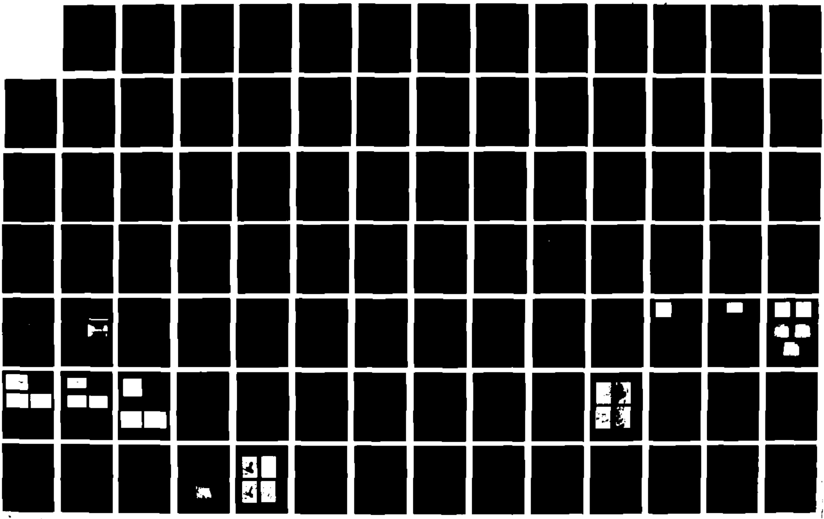
Results and Analysis

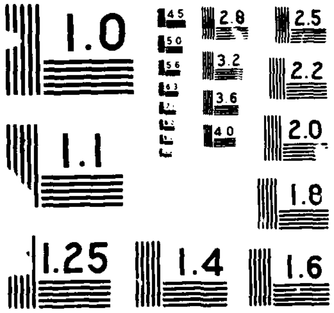
Due to limitations of time and resources, only a few of the participants were able to complete the entire benchmark exercise and test it on all five of the data sets. In almost every case, there was some disclaimer to the effect that a particular architecture could have shown better performance given more implementation time or resources. It was common for participants to underestimate the effort required to implement the benchmark, and several who had said they would provide timings were unable to complete even a portion of the task prior to the workshop. Despite requests to groups that did not attend the workshop that they submit belated results to be included in this report, not one new benchmark report has been received. Thus, the results presented here are those that were provided by the workshop participants. In a few cases, the results have been updated, corrected, or amended since they were originally presented.

Care must be taken in comparing these results. For example, no direct comparison should be made between results obtained from actual execution and those that were derived from simulation [Carpenter, 1987]. No matter how carefully a simulation is carried out, it is never as accurate as direct execution. Likewise, no comparison should be made between results from a partial implementation and a complete one. The complete implementation must account for overhead involved in the interactions between subtasks, and even for the fact that the program is significantly larger than for a partial implementation. Consider that a set of subtasks might appear to be much faster than their counterparts in a complete implementation simply because less paging is required to keep the code in memory. It is also unwise to directly compare the raw timings, even for similar architectures, without considering the differences in technology between systems. For example, a system that executes a portion of the benchmark in half the time of another is not necessarily architecturally superior if it also has a clock rate that is twice as high or if it has twice as many processors.

In addition to the technical problems involved in making direct comparisons, there are other considerations that must be kept in mind. For example, every participant expressed the view that given more time to tune their implementation, the results for their architecture would improve considerably. What is impressive in many cases is not the raw speed increase obtained, but the increase with respect to the amount of effort required to obtain it. While this has more to do with the tools available for developing software for an architecture than with the architecture itself, it is still important in evaluating the overall usefulness of the system. Another major consideration is the ratio of cost to performance, since many applications can afford to sacrifice a small amount of performance in order to reduce the cost of the implementation. In other applications, the size or weight or power consumption of a system may be of greater importance than all-out speed. One of the purposes of this exercise has been merely to assemble as much of this data as possible so that the performance results can be evaluated with respect to the requirements of each potential application of an architecture.

AD-A212 981 PROCEEDINGS OF IMAGE UNDERSTANDING WORKSHOP HELD IN 3/13
PALO ALTO CALIFORNIA O . (U) SCIENCE APPLICATIONS
INTERNATIONAL CORP MCLEAN VA L S BAUMAN MAY 89
UNCLASSIFIED N00014-86-C-0700 F/G 17/11 NL




1.0
4.5
5.0
5.6
6.3
2.8
3.2
3.6
4.0
2.5
2.2
2.0
1.1
1.8
1.25
1.4
1.6

Thus, in what follows, there is no single best architecture and there are no winners or losers. Each has its own unique merits and drawbacks, of which none are absolute. To play down the direct comparison of raw timings, the results for each architecture will be presented separately. The order of presentation is random, except that the sequential solution is presented first to provide a performance baseline, and then complete parallel implementations are presented, followed by partial implementations. Results that were based on theoretical estimations are not included in this report. The timings in all of the tables are in seconds, for the sake of consistency. Where a timing is zero, it indicates that the processing time was less than the resolution of the timing mechanism employed. Blanks in the tables indicate values that were omitted from the reports that were supplied by the implementors.

Sequential Solution

The sequential solution to the benchmark was developed in C on a Sun-3/160 workstation. The solution contains roughly 4600 lines of code, including comments. The implementation was designed for maximum portability and has been successfully recompiled on several different systems. The only portion that is system dependent is the actual result presentation step, which uses the graphics primitives provided for drawing on the workstation's screen. The implementation differs from the recommended method on the Connected Component Labelling step by using a standard sequential method for computing this well-defined function. The sequential method is designed to minimize array accesses and their corresponding index calculations, which is not a problem for array processors, but incurs a significant time penalty on a sequential machine.

Timings have been produced for the sequential code running on all five data sets, and on three different machine configurations. The configurations are a Sun-3/160 (a 16 MHz 68020 processor) with 8MB of RAM, a Sun-3/260 (a 25 MHz 68020) with 16MB of RAM, and a Sun-4/260 (a 16MHz SPARC processor) with 16MB of RAM. The extra RAM on the latter two machines did not affect performance, since the benchmark was able to run in 8MB without paging. The 3/260 was equipped with a Weitek floating-point co-processor, while the 3/160 used only the standard 68881 co-processor. These results have been corrected since the workshop, where some questions arose as to their validity due to a difference in the number of connected components extracted. It was determined that the original results were obtained with a faulty copy of the data set, and the problems vanished when the proper data was used. Table 5 shows the results for the Sun-3/160, Table 6 shows the Sun-3/260 results, and Table 7 gives the execution times for the Sun-4/260. The timings were obtained with the standard system clock utility which has a resolution of 20 milliseconds on the Sun-3 systems, and 10 milliseconds on the Sun-4.

Data Set	Sample		Test		Test2		Test3		Test4	
	User	System	User	System	User	System	User	System	User	System
Total	794.94	2.94	335.96	2.10	326.84	2.40	549.3	2.52	550.26	2.90
Overhead	4.02	1.06	4.06	0.88	4.50	1.14	4.60	1.04	4.58	0.94
Miscellaneous	2.24	0.04	2.18	0.04	2.16	0.06	2.12	0.02	2.10	0.02
Startup	0.02	0.00	0.04	0.00	0.02	0.04	0.00	0.02	0.02	0.00
Image input	0.60	0.68	0.58	0.54	1.32	0.78	1.50	0.74	1.42	0.66
Image output	0.24	0.30	0.30	0.28	0.06	0.24	0.06	0.24	0.08	0.26
Model input	0.92	0.04	0.96	0.02	0.94	0.02	0.92	0.02	0.96	0.00
Label connected components	27.40	0.38	27.46	0.36	28.12	0.28	27.86	0.36	27.88	0.36
Rectangles from intensity	6.42	0.08	4.00	0.14	4.34	0.04	5.36	0.08	5.10	0.24
Miscellaneous	2.06	0.06	1.84	0.02	1.94	0.02	1.94	0.02	1.92	0.06
Trace region boundary	0.52	0.02	0.28	0.02	0.38	0.00	0.42	0.00	0.38	0.06
K-curvature	1.62	0.00	0.80	0.00	0.82	0.00	1.22	0.00	1.10	0.00
K-curvature smoothing	1.26	0.00	0.62	0.00	0.70	0.00	0.96	0.00	1.02	0.02
First derivative	0.46	0.00	0.22	0.02	0.24	0.00	0.28	0.02	0.22	0.02
Zero-crossing detection	0.26	0.00	0.06	0.00	0.04	0.00	0.18	0.00	0.24	0.02

Final corner detection	0.20	0.00	0.16	0.02	0.18	0.02	0.28	0.02	0.16	0.04
Count corners	0.00	0.00	0.00	0.02	0.02	0.00	0.00	0.02	0.00	0.00
Convex hull	0.02	0.00	0.00	0.02	0.00	0.00	0.02	0.00	0.04	0.00
Test for right angles	0.00	0.00	0.02	0.02	0.00	0.00	0.04	0.00	0.00	0.00
Final rectangle hypothesis	0.02	0.00	0.00	0.00	0.02	0.00	0.02	0.00	0.02	0.02
Median filter	246.06	0.60	118.62	0.26	92.58	0.28	90.70	0.22	90.66	0.24
Sobel	135.3	0.18	133.14	0.16	135.92	0.18	135.12	0.16	135.14	0.28
Initial graph match	24.4	0.06	24.94	0.06	26.02	0.02	68.30	0.14	67.48	0.14
Match data rectangles	0.14	0.00	0.10	0.02	0.08	0.02	0.26	0.04	0.24	0.00
Match links	0.22	0.00	0.06	0.00	0.08	0.00	0.74	0.00	0.58	0.02
Create probe list	0.00	0.00	0.00	0.00	0.00	0.00	0.02	0.00	0.02	0.00
Partial match	24.04	0.06	24.78	0.04	25.86	0.00	67.28	0.10	66.64	0.12
Match strength probes	24.02	0.06	24.74	0.02	25.82	0.00	66.64	0.10	65.82	0.12
Window selection	0.02	0.00	0.02	0.00	0.00	0.00	0.12	0.02	0.10	0.02
Classification and count	24.0	0.06	24.72	0.02	25.82	0.00	66.50	0.06	65.70	0.08
Match extension	326.54	0.50	11.46	0.12	18.72	0.20	202.58	0.32	204.68	0.44
Match strength probes	72.88	0.10	3.28	0.00	5.80	0.06	47.82	0.06	42.00	0.06
Window selection	0.08	0.00	0.00	0.00	0.00	0.00	0.08	0.02	0.10	0.00
Classification and count	72.80	0.10	3.28	0.00	5.80	0.06	47.72	0.02	41.88	0.06
Hough probes	253.32	0.38	8.16	0.12	12.84	0.12	153.76	0.22	161.98	0.36
Window selection	0.00	0.00	0.00	0.00	0.00	0.00	0.08	0.02	0.02	0.02
Hough transform	252.20	0.36	8.10	0.12	12.78	0.12	151.86	0.16	160.34	0.28
Edge peak detection	1.08	0.02	0.06	0.00	0.06	0.00	1.76	0.00	1.54	0.02
Rectangle parameter update	0.04	0.00	0.00	0.00	0.00	0.00	0.04	0.02	0.04	0.00
Result presentation	24.80	0.00	12.28	0.04	16.64	0.02	14.78	0.00	14.74	0.02
Best match selection	0.00	0.00	0.00	0.00	0.00	0.00	0.02	0.00	0.00	0.00
Image generation	24.80	0.00	12.28	0.04	16.64	0.02	14.76	0.00	14.74	0.02
Statistics										
Connected components	134		35		34		114		100	
Right angles extracted	126		99		92		210		197	
Rectangles detected	25		21		16		42		39	
Depth pixels > threshold	21256		14542		12898		18584		18825	
Elements on initial probe list	381		19		27		400		249	
Hough probes	55		3		5		97		93	
Initial match strength probes	28		20		15		142		142	
Extension mat. str. probes	60		3		5		110		97	
Models remaining	2		1		1		2		1	
Model selected	10		1		5		7		8	
Average match strength		0.64		0.96		0.94		0.84		0.88
Translated to	151,240		256,256		257,255		257,255		257.255	
Rotated by (degrees)	85		359		114		22		22	

Table 5: Sun-3/160 Results

Data Set	Sample		Test		Test2		Test3		Test4	
	User	System	User	System	User	System	User	System	User	System
Total	293.42	5.96	130.48	2.06	116.96	2.56	191.38	3.38	192.38	3.20
Overhead	2.26	0.66	2.46	0.58	2.76	0.68	2.50	0.94	2.72	0.72

Miscellaneous	1.28	0.00	1.24	0.00	1.24	0.02	1.22	0.02	1.22	0.00
Startup	0.02	0.00	0.00	0.00	0.00	0.02	0.00	0.04	0.02	0.00
Image input	0.30	0.50	0.50	0.50	1.00	0.48	0.76	0.72	0.92	0.54
Image output	0.18	0.14	0.26	0.08	0.06	0.16	0.06	0.14	0.08	0.18
Model input	0.48	0.02	0.46	0.00	0.46	0.00	0.46	0.02	0.48	0.00
Label connected components	14.14	0.38	14.20	0.26	14.10	0.36	14.46	0.12	14.40	0.26
Rectangles from intensity	3.60	0.14	2.36	0.02	2.44	0.04	3.12	0.04	2.90	0.08
Miscellaneous	1.28	0.02	1.12	0.00	1.22	0.02	1.26	0.00	1.08	0.00
Trace region boundary	0.28	0.02	0.20	0.00	0.18	0.00	0.14	0.02	0.26	0.04
K-curvature	0.82	0.02	0.44	0.02	0.42	0.00	0.68	0.00	0.48	0.02
K-curvature smoothing	0.78	0.02	0.26	0.00	0.42	0.02	0.50	0.00	0.56	0.00
First derivative	0.20	0.02	0.16	0.00	0.10	0.00	0.18	0.00	0.26	0.00
Zero-crossing detection	0.02	0.02	0.04	0.00	0.06	0.00	0.18	0.00	0.14	0.00
Final corner detection	0.20	0.00	0.12	0.00	0.04	0.00	0.18	0.00	0.04	0.00
Count corners	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
Convex hull	0.02	0.00	0.00	0.00	0.00	0.00	0.00	0.20	0.04	0.00
Test for right angles	0.00	0.00	0.02	0.00	0.00	0.00	0.00	0.00	0.04	0.00
Final rectangle hypothesis	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.02
Median filter	112.50	1.20	59.86	0.42	42.64	0.46	42.64	0.34	42.72	0.54
Sobel	38.96	2.04	38.12	0.38	37.90	0.44	38.02	0.74	38.14	0.42
Initial graph match	6.10	0.06	6.06	0.02	6.38	0.20	17.02	0.30	16.80	0.14
Match data rectangles	0.08	0.00	0.06	0.00	0.04	0.00	0.14	0.02	0.12	0.00
Match links	0.10	0.00	0.04	0.00	0.04	0.00	0.30	0.00	0.26	0.00
Create probe list	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
Partial match	5.92	0.06	5.96	0.02	6.30	0.20	16.58	0.28	16.42	0.14
Match strength probes	5.90	0.06	5.94	0.02	6.30	0.20	16.34	0.22	16.04	0.14
Window selection	0.00	0.00	0.00	0.00	0.00	0.00	0.10	0.02	0.02	0.00
Classification and count	5.90	0.06	5.94	0.02	6.30	0.18	16.24	0.18	16.02	0.10
Match extension	109.18	1.28	3.78	0.14	6.02	0.22	69.32	0.76	70.42	0.74
Match strength probes	17.54	0.02	0.78	0.00	1.40	0.00	11.60	0.06	10.20	0.10
Window selection	0.04	0.00	0.00	0.00	0.00	0.00	0.04	0.00	0.04	0.00
Classification and count	17.50	0.02	0.78	0.00	1.40	0.00	11.56	0.06	10.16	0.08
Hough probes	91.44	1.26	3.00	0.12	4.62	0.20	57.30	0.66	59.80	0.64
Window selection	0.04	0.00	0.00	0.00	0.00	0.00	0.04	0.02	0.02	0.00
Hough transform	90.64	1.24	2.98	0.12	4.60	0.20	56.40	0.64	59.00	0.62
Edge peak detection	0.76	0.02	0.02	0.00	0.02	0.00	0.82	0.00	0.78	0.02
Rectangle parameter update	0.00	0.00	0.00	0.00	0.00	0.00	0.04	0.00	0.00	0.00
Result presentation	6.68	0.00	3.64	0.00	4.72	0.00	4.30	0.20	4.28	0.02
Best match selection	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
Image generation	6.68	0.00	3.64	0.00	4.72	0.00	4.30	0.02	4.28	0.02
Statistics										
Connected components	134		35		34		114		100	
Right angles extracted	126		99		92		210		197	
Rectangles detected	25		21		16		42		39	
Depth pixels > threshold	21256		14542		12898		18584		18825	
Elements on initial probe list	381		19		27		400		249	
Hough probes	55		3		5		97		93	
Initial match strength probes	28		20		15		142		142	
Extension mat. str. probes	60		3		5		110		97	
Models remaining	2		1		1		2		1	
Model selected	10		1		5		7		8	

Average match strength	0.64	0.96	0.94	0.84	0.88
Translated to	151,240	256,256	257,255	257,255	257,255
Rotated by (degrees)	85	359	114	22	22

Table 6: Sun-3/260 Results

Data Set	Sample		Test		Test2		Test3		Test4	
	User	System	User	System	User	System	User	System	User	System
Total	117.21	3.80	40.19	2.45	38.88	2.06	78.41	2.64	80.15	2.69
Overhead	2.49	1.85	2.34	1.58	2.43	1.36	2.62	1.46	2.66	1.45
Miscellaneous	1.23	1.20	1.17	0.81	1.24	0.70	1.45	0.77	1.43	0.74
Startup	0.02	0.03	0.00	0.05	0.03	0.02	0.01	0.05	0.01	0.06
Image input	0.33	0.48	0.27	0.58	0.33	0.47	0.35	0.46	0.38	0.47
Image output	0.10	0.11	0.12	0.10	0.05	1.11	0.05	0.10	0.09	0.09
Model input	0.52	0.02	0.50	0.02	0.50	0.04	0.50	0.04	0.49	0.04
Label connected components	4.39	0.35	4.29	0.27	4.31	0.23	4.36	0.26	4.33	0.28
Rectangles from intensity	1.01	0.09	0.68	0.00	0.67	0.04	0.86	0.10	0.87	0.04
Miscellaneous	0.31	0.05	0.32	0.00	0.27	0.02	0.33	0.05	0.32	0.02
Trace region boundary	0.06	0.01	0.04	0.00	0.04	0.01	0.04	0.00	0.03	0.00
K-curvature	0.21	0.00	0.05	0.00	0.11	0.00	0.08	0.00	0.08	0.00
K-curvature smoothing	0.22	0.00	0.16	0.00	0.15	0.00	0.21	0.01	0.22	0.00
First derivative	0.12	0.00	0.09	0.00	0.06	0.00	0.14	0.00	0.08	0.00
Zero-crossing detection	0.04	0.01	0.01	0.00	0.00	0.01	0.02	0.00	0.04	0.00
Final corner detection	0.04	0.01	0.01	0.00	0.03	0.00	0.02	0.02	0.06	0.00
Count corners	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.02
Convex hull	0.00	0.00	0.00	0.00	0.00	0.00	0.01	0.01	0.00	0.00
Test for right angles	0.00	0.00	0.00	0.00	0.00	0.00	0.01	0.00	0.01	0.00
Final rectangle hypothesis	0.01	0.01	0.00	0.00	0.00	0.00	0.00	0.00	0.03	0.00
Median filter	30.33	0.20	14.47	0.17	11.14	0.16	11.16	0.14	11.15	0.19
Sobel	11.21	0.95	11.26	0.17	11.17	0.10	11.11	0.30	11.15	0.30
Initial graph match	3.41	0.01	3.36	0.10	3.53	0.01	10.01	0.09	9.83	0.11
Match data rectangles	0.03	0.00	0.00	0.03	0.02	0.00	0.05	0.01	0.04	0.02
Match links	0.07	0.00	0.01	0.01	0.02	0.00	0.22	0.01	0.18	0.00
Create probe list	0.03	0.00	0.02	0.00	0.01	0.00	0.12	0.00	0.12	0.01
Partial match	3.28	0.01	3.33	0.06	3.48	0.01	9.62	0.07	9.49	0.08
Match strength probes	3.27	0.10	3.33	0.60	3.4[illegible]	0.01	9.44	0.07	9.30	0.08
Window selection	0.00	0.00	0.01	0.00	0.00	0.00	0.01	0.00	0.04	0.01
Classification and count	3.15	0.00	3.23	0.06	3.38	0.01	8.85	0.05	8.65	0.02
Match extension	60.98	0.26	2.06	0.12	3.35	0.08	36.18	0.23	38.10	0.26
Match strength probes	9.89	0.02	0.45	0.00	0.79	0.00	6.63	0.02	6.06	0.02
Window selection	0.00	0.00	0.00	0.00	0.00	0.00	0.03	0.01	0.01	0.00
Classification and count	9.60	0.00	0.44	0.00	0.78	0.00	6.12	0.00	5.56	0.02
Hough probes	50.99	0.21	1.61	0.12	2.56	0.08	29.32	0.20	31.77	0.22
Window selection	0.03	0.00	0.00	0.00	0.01	0.00	0.09	0.01	0.07	0.00
Hough transform	50.65	0.12	1.60	0.11	2.54	0.07	28.86	0.08	31.32	0.12
Edge peak detection	0.15	0.00	0.01	0.00	0.01	0.00	0.24	0.02	0.21	0.00
Rectangle parameter update	0.03	0.01	0.00	0.00	0.00	0.00	0.03	0.01	0.06	0.00
Result presentation	3.37	0.01	1.67	0.00	2.24	0.00	2.07	0.00	2.02	0.00
Best match selection	0.06	0.00	0.02	0.00	0.02	0.00	0.10	0.00	0.04	0.00
Image generation	3.31	0.01	1.65	0.00	2.22	0.00	1.97	0.00	1.98	0.00
Statistics										

Connected components	134	35	34	114	100
Right angles extracted	126	99	92	210	197
Rectangles detected	25	21	16	42	39
Depth pixels > threshold	21254	14531	12892	18579	18822
Elements on initial probe list	381	19	27	389	248
Hough probes	55	3	5	93	92
Initial match strength probes	28	20	15	142	142
Extension mat. str. probes	60	3	5	105	97
Models remaining	2	1	1	2	1
Model selected	10	1	5	7	8
Average match strength	0.64	0.96	0.94	0.84	0.88
Translated to	151,240	256,256	257,255	257,255	257,255
Rotated by (degrees)	85	359	114	22	22

Table 7: Sun-4/260 Results

Alliant FX-80 Solution

The Alliant FX-80 consists of up to eight computational elements and up to twelve I/O processors that share a physical memory through a sophisticated combination of caches, busses and an interconnection network. The computational elements communicate with the shared memory via the interconnection network which links them to a pair of special purpose caches that in turn access the memory over a bus that is shared with the I/O processor caches. The FX-80 differs from the older FX-8 primarily in that the computational elements are significantly faster.

Alliant was able to implement the benchmark on the FX-80 in roughly one programmer-week. The programmer who built the implementation had no experience in vision and, in many cases, did not even bother to learn how the benchmark code works. The implementation was done by rewriting the system dependent section to use the available graphics hardware, compiling the code with Alliant's vectorizing and globally optimizing C compiler, using a profiling tool to determine the portions of the code that used the greatest percentage of CPU time, inserting compiler directives in the form of comments to break implicit dependencies in four sections of the benchmark, and recompiling the new version of the code. Alliant provided results for five configurations of the FX-80, with 1, 2, 4, 6, and 8 computational elements. In order to save space, only two of the configurations are represented here. Table 8 shows the execution times for a single FX-80 computational element, and Table 9 shows the results for an FX-80 with eight elements. Another point that was noted by Alliant is that the C compiler is a new product and does not yet provide as great a degree of optimization as their FORTRAN compiler (a difference of up to 50% in some cases). They expect to see significantly better performance with later releases of the product.

Data Set	Sample		Test		Test2		Test3		Test4	
	User	System	User	System	User	System	User	System	User	System
Total	204.858	2.531	102.700	1.861	93.311	1.828	136.759	3.049	139.130	3.032
Overhead	7.968	0.776	7.925	0.777	7.897	0.775	7.900	0.764	7.895	0.763
Miscellaneous	0.627	0.030	0.585	0.033	0.559	0.033	0.554	0.030	0.554	0.031
Startup	0.030	0.031	0.029	0.033	0.029	0.031	0.029	0.032	0.029	0.029
Image input	5.692	0.515	5.691	0.051	5.691	0.505	5.697	0.509	5.690	0.504
Image output	1.039	0.175	1.039	0.179	1.038	0.183	1.039	0.171	1.040	0.177
Model input	0.580	0.021	0.058	0.017	0.580	0.018	0.580	0.017	0.580	0.019
Label connected components	16.917	0.268	16.830	0.258	16.800	0.253	16.948	0.247	16 930	0.259
Rectangles from intensity	2.760	0.590	1.791	0.267	1.874	0.252	2.312	0.681	2.286	0.643
Miscellaneous	1.005	0.231	0.928	0.097	0.931	0.094	0.986	0.255	0.983	0.239
Trace region boundary	0.312	0.078	0.172	0.021	0.183	0.019	0.255	0.062	0.221	0.054
K-curvature	0.592	0.037	0.287	0.017	0.308	0.017	0.438	0.045	0.432	0.045

K-curvature smoothing	0.362 0.037	0.176 0.018	0.188 0.017	0.269 0.045	0.264 0.044
First derivative	0.158 0.037	0.077 0.017	0.082 0.016	0.119 0.045	0.117 0.043
Zero-crossing detection	0.170 0.037	0.076 0.017	0.099 0.017	0.135 0.045	0.133 0.043
Final corner detection	0.135 0.042	0.060 0.022	0.069 0.022	0.103 0.051	0.101 0.049
Count corners	0.006 0.037	0.003 0.017	0 002 0.017	0.007 0.044	0.006 0.042
Convex hull	0.013 0.026	0.006 0.017	0.006 0.017	0.015 0.042	0.015 0.040
Test for right angles	0.006 0.013	0.005 0.011	0.004 0.009	0.009 0.022	0.008 0.021
Final rectangle hypothesis	0.003 0.013	0.003 0.011	0.002 0.009	0.006 0.022	0.005 0.021
Median filter	77.294 0.170	43.652 0.160	31.886 0.163	31.919 0.154	31.880 0.166
Sobel	26.147 0.001	26.079 0.001	26.063 0.001	26.128 0.001	26.129 0.001
Initial graph match	2.458 0.088	2.397 0.063	2.569 0.055	7.117 0.368	7.011 0.373
Match data rectangles	0.067 0.023	0.051 0.012	0.046 0.014	0.129 0.047	0.111 0.041
Match links	0.067 0.002	0.024 0.004	0.022 0.004	0.262 0.013	0.214 0.023
Create probe list	0.002 0.001	0.002 0.001	0.002 0.001	0.005 0.001	0.006 0.003
Partial match	2.321 0.062	2.320 0.046	2.499 0.036	6.722 0.307	6.680 0.307
Match strength probes	2.305 0.045	2.303 0.032	2.486 0.024	6.502 0.228	6.429 0.229
Window selection	0.009 0.032	0.003 0.011	0.002 0.008	0.020 0.076	0.020 0.077
Classification and count	2.299 0.015	2.298 0.011	2.482 0.008	6.471 0.076	6.397 0.076
Match extension	68.025 0.385	2.149 0.083	3.817 0.091	42.243 0.600	44.806 0.584
Match strength probes	7.139 0.096	0.311 0.005	0.568 0.008	4.600 0.168	4.216 0.155
Window selection	0.009 0.032	0.000 0.002	0.001 0.003	0.15 0.056	0.014 0.052
Classification and count	7.125 0.032	0.310 0.002	0.566 0.003	4.576 0.056	4.193 0.052
Hough probes	60.754 0.202	1.833 0.068	3.241 0.071	37.330 0.301	40.320 0.312
Window selection	0.008 0.030	0.001 0.002	0.001 0.003	0.014 0.051	0.014 0.051
Hough transform	60.259 0.082	1.806 0.061	3.210 0.061	36.650 0.097	39.604 0.110
Edge peak detection	0.474 0.031	0.026 0.002	0.030 0.003	0.642 0.050	0.681 0.050
Rectangle parameter update	0.008 0.030	0.000 0.002	0.001 0.003	0.015 0.051	0.014 0.051
Result presentation	3.269 0.002	1.860 0.002	2.388 0.002	2.177 0.002	2.174 0.002
Best match selection	0.003 0.001	0.001 0.001	0.001 0.001	0.004 0.001	0.002 0.001
Image generation	3.266 0.001	1.859 0.001	2.387 0.001	2.174 0.001	2.172 0.001
Statistics					
Connected components	134	35	34	114	100
Right angles extracted	126	99	92	210	197
Rectangles detected	25	21	16	42	39
Depth pixels > threshold	21266	14542	12888	18572	18813
Elements on initial probe list	374	19	27	389	248
Hough probes	55	3	5	93	92
Initial match strength probes	28	20	15	142	142
Extension mat. str. probes	60	3	5	105	97
Models remaining	2	1	1	2	1
Model selected	10	1	5	7	8
Average match strength	0.65	0.96	0.94	0.84	0.88
Translated to	151,240	256,256	257,255	257,255	257,255
Rotated by	85	359	114	22	22

Table 8: Alliant FX-80 Single Processor Results

Data Set	Sample		Test		Test2		Test3		Test4	
	User	System	User	System	User	System	User	System	User	System
Total	57.177	2.935	31.056	2.082	30.872	2.043	50.357	3.577	50.153	3.467
Overhead	7.940	0.847	7.903	0.825	7.897	0.813	7.891	0.820	7.899	0.822
Miscellaneous	0.601	0.042	0.558	0.039	0.558	0.039	0.553	0.041	0.560	0.058
Startup	0.030	0.056	0.029	0.047	0.029	0.042	0.029	0.043	0.029	0.033
Image input	5.690	0.549	5.695	0.541	5.691	0.532	5.690	0.542	5.690	0.536
Image output	1.039	0.173	1.040	0.172	1.038	0.177	1.039	0.173	1.039	0.173
Model input	0.580	0.023	0.580	0.021	0.580	0.017	0.580	0.017	0.580	0.017
Label connected components	6.930	0.295	6.864	0.272	6.849	0.270	6.979	0.273	6.992	0.272
Rectangles from intensity	2.776	0.686	1.799	0.314	1.882	0.295	2.329	0.785	2.309	0.751
Miscellaneous	1.010	0.277	0.931	0.120	0.934	0.113	0.994	0.303	0.990	0.290
Trace region boundary	0.312	0.084	0.172	0.023	0.183	0.022	0.227	0.071	0.224	0.063
K-curvature	0.594	0.042	0.287	0.020	0.308	0.019	0.438	0.051	0.433	0.049
K-curvature smoothing	0.364	0.042	0.176	0.019	0.189	0.019	0.270	0.052	0.267	0.050
First derivative	0.159	0.042	0.077	0.019	0.083	0.019	0.120	0.051	0.120	0.050
Zero-crossing detection	0.171	0.049	0.077	0.020	0.100	0.019	0.136	0.051	0.135	0.050
Final corner detection	0.136	0.048	0.060	0.028	0.070	0.025	0.103	0.057	0.130	0.055
Count corners	0.007	0.041	0.003	0.019	0.003	0.019	0.008	0.050	0.007	0.052
Convex hull	0.014	0.030	0.007	0.019	0.007	0.019	0.016	0.047	0.016	0.045
Test for right angles	0.006	0.016	0.005	0.013	0.004	0.010	0.010	0.025	0.009	0.023
Final rectangle hypothesis	0.004	0.015	0.003	0.013	0.002	0.010	0.007	0.026	0.005	0.023
Median filter	9.890	0.223	5.637	0.220	4.111	0.212	4.110	0.214	4.109	0.209
Sobel	3.798	0.001	3.789	0.001	3.787	0.001	3.795	0.001	3.795	0.001
Initial graph match	2.455	0.123	2.399	0.094	2.569	0.086	7.130	0.485	7.014	0.459
Match data rectangles	0.068	0.048	0.052	0.028	0.046	0.033	0.131	0.102	0.112	0.083
Match links	0.068	0.004	0.024	0.009	0.022	0.009	0.263	0.030	0.213	0.020
Create probe list	0.002	0.001	0.002	0.001	0.002	0.001	0.005	0.001	0.006	0.004
Partial match	2.317	0.070	2.322	0.055	2.499	0.043	6.732	0.351	6.682	0.351
Match strength probes	2.301	0.050	2.304	0.037	2.485	0.027	6.509	0.259	6.429	0.263
Window selection	0.004	0.017	0.004	0.012	0.002	0.009	0.023	0.087	0.025	0.087
Classification and count	2.294	0.017	2.298	0.012	2.482	0.009	6.473	0.085	6.390	0.087
Match extension	20.105	0.455	0.786	0.107	1.376	0.122	15.926	0.739	15.845	0.702
Match strength probes	7.121	0.111	0.311	0.006	0.567	0.009	4.609	0.195	4.219	0.185
Window selection	0.010	0.037	0.001	0.002	0.001	0.003	0.019	0.065	0.016	0.065
Classification and count	7.105	0.037	0.310	0.002	0.565	0.003	4.580	0.066	4.193	0.060
Hough probes	12.847	0.243	0.468	0.086	0.799	0.099	10.996	0.378	11.350	0.366
Window selection	0.008	0.033	0.001	0.002	0.001	0.003	0.014	0.057	0.014	0.057
Hough transform	12.353	0.110	0.441	0.078	0.767	0.086	10.315	0.151	10.629	0.140
Edge peak detection	0.472	0.034	0.026	0.002	0.030	0.003	0.645	0.057	0.682	0.057
Rectangle parameter update	0.009	0.033	0.000	0.002	0.001	0.003	0.013	0.056	0.014	0.057
Result presentation	3.265	0.002	1.859	0.002	2.382	0.002	2.178	0.002	2.173	0.002
Best match selection	0.003	0.001	0.001	0.001	0.001	0.001	0.004	0.001	0.002	0.00
Image generation	3.262	0.001	1.858	0.001	2.381	0.001	2.174	0.001	2.171	0.001
Statistics										
Connected components	134		35		34		114		100	
Right angles extracted	126		99		92		210		197	
Rectangles detected	25		21		16		42		39	
Depth pixels > threshold	21266		14542		12888		18572		18813	
Elements on initial probe list	374		19		27		389		248	

Hough probes	55	3	5	93	92
Initial match strength probes	28	20	15	142	142
Extension mat. str. probes	60	3	5	105	97
Models remaining	2	1	1	2	1
Model selected	10	1	5	7	8
Average match strength	0.65	0.96	0.94	0.84	0.88
Translated to	151,240	256,256	257,255	257,255	257,255
Rotated by	85	359	114	22	22

Table 9: Alliant FX-80 Results with Eight Processors

Image Understanding Architecture

The Image Understanding Architecture (IUA) is being built by the University of Massachusetts and Hughes Research Laboratories specifically to address the problem of supporting real-time, knowledge-based vision. The architecture consists of three different parallel processors that are arranged in a hierarchy that is tightly coupled by layers of dual-ported memory between the processors. The low-level processor in the hierarchy is a bit-serial, processor-per-pixel, SIMD, associative array. The intermediate-level processor is an MIMD array of 4096 16-bit digital signal processors that can communicate via an interconnection network. Each intermediate-level processor shares a dual-ported memory segment with 64 low-level processors. The high level is a multiprocessor that is designed to support AI processing and a blackboard model of communication through a global shared memory, which is dual-ported with a segment of the intermediate-level processor's memory. A detailed description of the architecture can be found in [Weems, 1989].

Because the architecture is still under construction, an instruction-level simulator was used to develop the benchmark implementation. The simulator is programmed in a combination of Forth and an assembly language which has a syntax that is similar to Ada or Pascal. The benchmark was developed over a period of about six months, but much of that time was spent in building basic library routines and additional tools that were generally required for any large programming task. A 1/64th scale version of the simulator (4096 low-level, 64 intermediate-level, and one high-level processor) runs on a Sun workstation, and was used to develop the initial benchmark implementation. The implementation was then transported to a full-scale IUA simulator running on a Sequent Symmetry multiprocessor. At the time of the Avon workshop, several errors remained in the full-scale implementation, but these have since been corrected. Table 10 presents the results from the IUA simulations with a resolution of one instruction time (0.1 microsecond). There are several points to note about these results. Because the processing of different steps can be overlapped in the different processing levels, the sum of the individual step timings does not always equal the total time for a segment of the benchmark. Some of the individual timings represent average execution times, since the intermediate level processing takes place asynchronously and individual processes can vary in their execution time. For example, the time for all of the match-strength probes is difficult to estimate since probes are created asynchronously and their processing is overlapped. However, the time for a step such as match extension takes into account the span of time required to complete all of the subsidiary match-strength probes. Lastly, it should be mentioned that the intermediate-level processor was greatly underutilized by the benchmark (only 0.2% of its processors were activated), and the high-level processor was not used at all. The low-level processor was also idle roughly 50% of the time while awaiting requests for top-down probes from the intermediate level.

Data Set	Sample	Test	Test2	Test3	Test4
Total	0.0844445	0.0455559	0.0455088	0.4180890	0.3978859
Overhead	0.0139435	0.0139435	0.0139435	0.0139435	0.0139435
Miscellaneous	0.0092279	0.0092279	0.0092279	0.0092279	0.0092279
Startup	0.0038682	0.0038682	0.0038682	0.0038682	0.0038682
Image input	0.0000020	0.0000020	0.0000020	0.0000020	0.0000020
Image output	0.0000020	0.0000020	0.0000020	0.0000020	0.0000020
Model input	0.0008302	0.0008302	0.0008302	0.0008302	0.0008302
Label connected components	0.0000596	0.0000596	0.0000596	0.0000596	0.0000596
Rectangles from intensity	0.0161694	0.0125489	0.0134704	0.0131378	0.0129635
Miscellaneous	0.0003227	0.0002421	0.0002010	0.0006216	0.0002421
Trace region boundary	0.0033792	0.0015472	0.0018672	0.0010912	0.0012832
K-curvature	0.0038256	0.0019936	0.0023136	0.0015376	0.0017296
K-curvature smoothing	0.0005525	0.0005525	0.0005525	0.0005525	0.0005525
First derivative	0.0003777	0.0003777	0.0003777	0.0003777	0.0003777
Zero-crossing detection	0.0000108	0.0000108	0.0000108	0.0000108	0.0000108
Final corner detection	0.0000118	0.0000118	0.0000118	0.0000118	0.0000118
Count corners	0.0000020	0.0000020	0.0000020	0.0000020	0.0000020
Convex hull	0.0036694	0.0019109	0.0015290	0.0025947	0.0026463
Test for right angles	0.0006122	0.0006009	0.0005906	0.0006421	0.0006421
Final rectangle hypothesis	0.0067877	0.0067877	0.0078821	0.0067877	0.0064229
Median filter	0.0005625	0.0005625	0.0005625	0.0005625	0.0005625
Sobel	0.0026919	0.0026919	0.0026919	0.0026919	0.0026919
Initial graph match	0.0121876	0.0076429	0.0066834	0.1124236	0.0822296
Match data rectangles	0.0029096	0.0015672	0.0013264	0.0134885	0.0106136
Match links	0.0088872	0.0056950	0.0049762	0.0985542	0.0712324
Create probe list	0.0000968	0.0001299	0.0001130	0.0009252	0.0008618
Partial match	0.0033786	0.0077033	0.0068704	0.1828976	0.1534418
Match strength probes	0.0009275	0.0011460	0.0012285	0.0025175	0.0212640
Window selection	0.0002100	0.0003000	0.0002700	0.0005700	0.0004800
Classification and count	0.0001043	0.0001490	0.0001341	0.0002831	0.0002384
Match extension	0.0300650	0.0017674	0.0024856	0.0899214	0.1277396
Match strength probes	0.0026500	0.0001146	0.0004095	0.0543250	0.0071766
Window selection	0.0006000	0.0000300	0.0000900	0.0012300	0.0016200
Classification and count	0.0002980	0.0000149	0.0000447	0.0006109	0.0008046
Hough probes	0.0068430	0.0003251	0.0005092	0.0084591	0.0109868
Window selection	0.0000675	0.0000045	0.0000090	0.0001755	0.0002385
Hough transform	0.0053010	0.0002223	0.0003036	0.0044499	0.0053477
Edge peak detection	0.0011745	0.0000783	0.0001566	0.0030537	0.0041499
Rectangle parameter update	0.0003000	0.0000200	0.0000400	0.0007800	0.0010600
Result presentation	0.0022826	0.0009452	0.0011944	0.0029768	0.0029766
Best match selection	0.0000404	0.0000403	0.0000405	0.0000406	0.0000397
Image generation	0.0022352	0.0009185	0.0011396	0.0029464	0.0029464
Statistics					
Connected components	134	35	34	114	100
Right angles extracted					
Rectangles detected	31	23	19	60	55
Depth pixels > threshold					
Elements on initial probe list					

Hough probes	44	5	8	84	100
Initial match strength probes	24	20	15	81	80
Extension mat. str. probes	20	1	3	41	54
Models remaining	3	1	1	2	1
Model selected	10	1	5	7	8
Average match strength	0.45	0.86	0.84	0.81	0.84
Translated to	151,240	256,256	257,255	257,255	257,255
Rotated by	85	359	113	23	23

Table 10: Image Understanding Architecture Results

Aspex ASP

The Associative String Processor (ASP) is being built by the University of Brunel and Aspex Ltd. in England [Lea, 1988]. It is designed as a general purpose processing array for implementation in wafer-scale technology. The processor consists of 262,144 processors arranged as 512 strings of 512 processors each. Each processor contains a 96-bit data register and a 5-bit activity register. A string consists of 512 processors linked by a communication network that is also tied to a data exchanger and a vector data buffer. The vector data buffers of the strings are linked through another data exchanger and data buffer to another communication network. One of the advantages of this arrangement is a high degree of fault tolerance. The system can be built with 1024 VLSI devices, or 128 ULSI devices, or 32 WSI devices. Estimated power consumption is 650 watts. The processor clock and instruction rate is 20 MHz. Architectural changes that would improve the benchmark performance include increasing the number of processors (improves performance on K-curvature, median filter, and Sobel), increasing the speed of the processors and communication links (linear speedup on all tasks), and adding a separate controller to each ASP substring (gives approximately an 18% increase overall).

Because the system is still under construction, a software simulator was used to implement and execute the benchmark. The benchmark was programmed in an extended version of Modula-2 over a period three months by two programmers, following a three month period of initial study of the requirements and development of a solution strategy. A Jarvis' March algorithm was substituted for the recommended Graham Scan method on the convex hull. Table 11 lists the benchmark results for the ASP. Timings were not provided for several of the steps in the model matching portion of the benchmark, possibly because a different method was used. Startup and model input times were not listed separately, perhaps because those operations are done outside of the simulation. The miscellaneous time under overhead accounts for the input and output of several intermediate images. The miscellaneous time under the section that extracts rectangles from the intensity image accounts for the output and subsequent input of data records for corners and rectangles. No indication was given of whether any data rearrangement took place as part of these I/O operations.

Data Set	Sample	Test	Test2	Test3	Test4
Total	0.1307200	0.0359600	0.0398100	0.1130700	0.1188200
Overhead	0.0008200	0.0008200	0.0008000	0.0008000	0.0008000
Miscellaneous	0.0002560	0.0002560	0.0002560	0.0002560	0.0002560
Startup					
Image input	0.0000512	0.0000512	0.0000512	0.0000512	0.0000512
Image output	0.0000512	0.0000512	0.0000512	0.0000512	0.0000512
Model input					
Label connected components	0.0392000	0.0228000	0.0228000	0.0348000	0.0313000
Rectangles from intensity	0.0033100	0.0029200	0.0028800	0.0031900	0.0033500
Miscellaneous	0.0000761	0.0000860	0.0000842	0.0000795	0.0000734

Trace region boundary	0.0000047	0.0000047	0.0000047	0.0000047	0.0000047
K-curvature	0.0007800	0.0007800	0.0007800	0.0007800	0.0007800
K-curvature smoothinc	0.0004500	0.0004500	0.0004500	0.0004500	0.0004500
First derivative	0.0000320	0.0000320	0.0000320	0.0000320	0.0000320
Zero-crossing de :ction	0.0000045	0.0000045	0.0000045	0.0000045	0.0000045
Final corne: detection	0.0000018	0.0000018	0.0000018	0.0000018	0.0000018
Count :orners	0.0000400	0.0000380	0.0000380	0.0000530	0.0000380
Convex hull	0.0003300	0.0002820	0.0002820	0.0003300	0.0003300
ıest for right angles	0.0008800	0.0008400	0.0008400	0.0009500	0.0009200
Final rectangle hypothesis	0.0004500	0.0003800	0.0002900	0.0007600	0.0007000
Median filter	0.0007200	0.0007200	0.0005100	0.0006100	0.0005100
Sobel	0.0006240	0.0006240	0.0006240	0.0006800	0.0006240
Initial graph match	0.0000090	0.0000090	0.0000090	0.0000090	0.0000080
Match data rectangles					
Match links					
Create probe list					
Partial match					
Match strength probes					
Window selection	0.0001200	0.0001320	0.0001080	0.0005500	0.0006400
Classification and count	0.0009500	0.0008850	0.0008650	0.0015400	0.0016000
Match extension	0.0835200	0.0001470	0.0001400	0.0002650	0.0002590
Match strength probes					
Window selection	0.0003000	0.0000240	0.0000360	0.0009200	0.0009800
Classification and count	0.0030000	0.0004050	0.0003520	0.0047200	0.0054500
Hough probes					
Window selection	0.0002880	0.0000240	0.0000360	0.0005800	0.0007300
Hough transform	0.0790000	0.0054000	0.0104000	0.0610000	0.0690000
Edge peak detection	0.0007700	0.0000640	0.0000990	0.0015400	0.0017600
Rectangle parameter update	0.0002160	0.0000090	0.0000100	0.0002340	0.0002360
Result presentation	0.0008500	0.0004400	0.0004700	0.0004700	0.0010300
Best match selection	0.0000250	0.0000150	0.0000150	0.0000280	0.0000150
Image generation	0.0007200	0.0003200	0.0003500	0.0008400	0.0009100
Statistics					
Connected components	34	33	113	99	
Right angles extracted	99	92	210	197	
Rectangles detected	21	16	42	39	
Depth pixels > threshold	14533	12891	18582	18817	
Elements on initial probe list					
Hough probes	3	5	97	93	
Initial match strength probes	20	15	142	142	
Extension mat. str. probes	3	5	110	97	
Models remaining	1	1	2	1	
Model selected	1	5	7	8	
Average match strength	0.96	0.93	0.84	0.87	
Translated to	256,256	257,255	257,255	257,255	
Rotated by	359	114	22	22	

Table 11: Aspex ASP Results

Sequent Symmetr', 81

The Sequent Computer Systems Symmetry 81 multiprocessor consists of multiple Intel 80386 microprocessors, running at 16.5 MHz, connected via a shared bus to a large shared memory. The particular configuration used to obtain these results included 12 processors (one of which is reserved by the system), each with an 80387 math coprocessor, and 96 MB of shared memory. The test system also contained the older A-model caches, which induce a considerably greater level of traffic on 'he shared bus than the newer B-model caches. An improvement of 30 to 50 percent in thc overall performance is possible with the new caching system. Sequent was to have provided timings for a system with the improved cache, but they have not yet done so. The timings presented in Table 12 were obtained by the benchmark devclopers at UMass as part of their effort to ensure the portability of the benchmark to different systems.

About a month was spent developing the parallel implementation for the Sequent. The programmer who did the work was famil:ar with the benchmark, but had no previous experience with the Sequent system. Part of the development period was spent back-porting modifications to the sequential version of the benchmark in order to enhance its portability. The low-level tasks were directly converted to a parallel implementation by dividing the data sets among the processors in a manner that completely avoided 'vrite-contention. About half of the development time was spent adding the appropriate data locking mechanisms to the model-matching portion of the benchmark, and resolving problems with timing and race conditions. It was only possible to obtain timings for the major steps in the benchmark, because thc Sequent operating system does not provide facilities for accurately timing individual child processes. The benchmark was run on configurations of from one to eleven processors, with the optimum time being obtained with eight or nine processors. Additional processors resulted in an overall reduction in performance, which was due to a combination of factors. As the data sets were divided among more processors, the ratio of processing time to task creation overhead decreased so that the latter came to dominate the time on some tasks. We also believe that some of the tasks reached the saturation point of the shared bus at about eight or nine processors since one run that was observed on a B-model cache system showed performance to improve with more processors. The table shows the performance obtained for a single processor running the sequential version of the benchmark, to provide a comparison baseline, and the performance on the optimum number of proccssors for each data set.

Data Set	Sample		Test		Test2		Test3		Test4	
	Single	Eight	Single	Eight	Single	Nine	Single	Eight	Singie	Nine
Total	889.66	251.33	300.34	73.88	282.71	77.87	562.15	174.96	578.14	139.72
Overhead	5.84	6.00	5.57	5.93	5.62	5.87	5.75	5.86	5.65	5.90
System time	3.60	9.40	2.00	5.40	2.10	6.40	2.80	7.60	2.90	8.80
Label conn. components	19.27	12.68	19.34	15.83	19.29	16.01	19.60	16.84	19.58	16.89
Rectangles from intensity	4.18	1.45	2.62	0.92	2.74	1.92	3.42	1.42	3.38	1.89
Median filter	239.24	31.00	114.12	15.25	85.81	11.08	85.83	11.45	85.79	11.11
Sobel	110.89	15.00	113.21	15.46	110.80	14.83	110.84	15.20	110.81	14.73
Initial graph match	18.52	3.08	18.53	3.76	19.90	4.35	52.53	7.21	51.63	7.17
Match data rectangles	0.17	0.04	0.11	0.03	0.09	0.03	0.26	0.13	0.22	0.06
Match links	0.19	0.24	0.06	0.20	0.06	0.65	0.74	0.29	0.59	0.78
Create probe list	0.01	0.00	0.01	0.01	0.01	0.01	0.01	0.01	0.01	0.01
Partial match	18.15	2.80	18.35	3.52	19.74	3.66	51.52	6.78	50.81	6.32
Match extension	470.90	161.34	16.16	5.97	24.08	9.38	271.07	103.99	288.21	69.10
Result presentation	20.82	20.78	10.80	10.76	14.47	14.43	13.11	12.99	13.09	12.93

Table 12: Sequent Symmetry 81 Results

Warp

The CMU Warp is a systolic array consisting of ten high speed floating point units in a linear configuration [Kung, 1984]. Processing in the Warp is directed by a host processor, such as the Sun-3/60 workstation that was used in executing the benchmark. The benchmark implementation was programmed by one person in two weeks, using a combination of the original C implementation and subroutines written in Apply and W2. The objective of the implementation was to obtain the best overall time, rather than the best time for each task. While it would seem that the latter guarantees the former, consider that the Warp and its host can work in parallel on different portions of a problem. Thus, even though the Warp could perform a step in one second that requires four seconds on the host, it is better to let the host do the processing if it would otherwise sit idle while the Warp is computing. Thus the Warp implementation of the benchmark exploits both the tightly-coupled parallelism of the Warp array, and the loosely-coupled task-level parallelism present in the benchmark.

Table 13 lists the results for the Warp. Timings were not provided for a few of the steps, but the totals include all of the processing time. The Miscellaneous category under Overhead is the time required for downloading code to the Warp array at various stages of the processing. A figure for the total system time was provided, rather than a breakdown of system time by task. The overall Total includes the system time, which is listed on the line below the Total. Note that sums of the times for the individual steps will not equal the Total time because of the task-level parallelism that was used.

Data Set	Sample	Test	Test2	Test3	Test4
Total	43.60	20.30	22.30	58.10	55.30
System	3.00	2.30	2.50	4.30	4.90
Overhead					
Miscellaneous	3.56	2.24	2.30	5.52	7.30
Startup	5.76	6.04	5.96	5.88	6.00
Image input	3.52	3.72	5.40	5.34	5.34
Image output					
Model input	1.30	1.18	1.02	1.08	1.06
Label connected components	3.98	4.04	4.60	4.54	4.56
Rectangles from intensity					
Miscellaneous					
Trace region boundary					
K-curvature	3.14	2.24	2.20	2.272	2.54
K-curvature smoothing	1.38	0.64	0.78	0.98	0.90
First derivative	0.42	0.24	0.28	0.34	0.40
Zero-crossing detection	0.32	0.06	0.12	0.14	0.22
Final corner detection	0.16	0.10	0.12	0.22	0.20
Count corners	0.02	0.02	0.04	0.06	0.06
Convex hull	0.02	0.00	0.02	0.08	0.06
Test for right angles	0.00	0.00	0.02	0.02	0.02
Final rectangle hypothesis	0.04	0.00	0.02	0.02	0.04
Median filter	10.70	8.70	1.38	1.40	2.00
Sobel	0.48	0.48	0.72	0.94	0.92
Initial graph match	0.42	0.24	0.22	1.22	1.38
Match data rectangles	0.20	0.16	0.16	0.40	0.68
Match links	0.22	0.08	0.06	0.82	0.70
Create probe list					
Partial match					
Match strength probes					

Window selection					
Classification and count					
Match extension	24.80	3.64	4.58	38.60	41.20
Match strength probes	9.10	2.64	2.86	13.60	13.50
Window selection	0.02	0.02	0.02	0.24	0.18
Classification and count	9.00	2.56	2.82	13.20	13.10
Hough probes	15.30	0.96	1.68	23.30	25.80
Window selection	0.02	0.00	0.02	0.12	0.06
Hough transform	12.80	0.88	1.44	19.30	20.00
Edge peak detection	2.38	0.08	0.22	3.80	5.58
Rectangle parameter update	0.02	0.00	0.00	0.00	0.08
Result presentation	2.60	2.26	2.52	2.24	2.26
Best match selection	0.02	0.00	0.00	0.02	0.02
Image generation	2.54	2.20	2.46	2.16	2.18
Statistics					
Total match strength probes	91	23	20	247	239
Hough probes	58	3	5	97	95

Table 13: Results for the Warp

Connection Machine

The Thinking Machines Connection Machine model CM-2 is a data-parallel array of bit-serial processors that are linked by an N-dimensional hypercube router network [Hillis, 1986]. In addition, for every 32 of the bit-serial processors, a 32-bit floating-point coprocessor is provided. Connectior Machines are available in configurations of 8192, 16384, 32768, and 65536 processing elements. Results were provided for direct execution on the three smaller configurations, and extrapolated to the largest configuration. The development team at Thinking Machines spent about three programmer months converting the low-level portion of the benchmark into 2600 lines of *LISP, which is a data-parallel extension to Common LISP. There was not enough time to implement the intermediate and top-down processing portions of the benchmark before the workshop, and other projects have taken priority over completing the benchmark since then. However, there was also some concern as to whether the Connection Machine would be the best vehicle for implementing the other portions, since they are more concerned with task parallelism than data parallelism. It was suggested that if the model data base included several thousand models to be matched, then an appropriate method might be found to take advantage of the Connection Machine's capabilities.

Table 14 summarizes the results for the Connection Machine on the low-level portion of the benchmark, with times rounded to two significant digits (as provided by Thinking Machines). A 32K-processor CM-2 with a Data Vault disk system and a Sun-4 host processor was used to obtain the results. The results that were supplied were for only one data set, and did not indicate which one was used. It is interesting to note that several of the tasks saw little speedup with the larger configurations of the Connection Machine. Those tasks involved a collection of contour values that had been mapped into 16K virtual processors, which are enough to operate on all of the contour points in parallel, and so there was no advantage in using more physical processors than virtual processors. It was suggested that the Connection Machine might thus be used to process the contours for several images at once in order to make use of the larger number of processors. On the other hand, for those tasks that are pixel oriented, 256K virtual processors were used and therefore a proportional speedup can be observed as the number of processors increases.

Configuration	8K	16K	32K	64K
Total (low level tasks only)	1.26	0.91	0.71	0.63
Overhead				
Miscellaneous				
Startup	0.10	0.10	0.10	0.1C
Image input	0.155	0.155	0.155	0.155
Image output				
Model input				
Label connected components	0.34	0.21	0.14	0.10
Rectangles from intensity				
Miscellaneous				
Trace region boundary	0.44	0.30	0.23	0.17
K-curvature	0.019	0.019	0.018	0.018
K-curvature smoothing	0.0056	0.0055	0.0062	0.0055
First derivative	0.00038	0.00037	0.00037	0.00037
Zero-crossing detection	0.00021	0.00020	0.00019	0.00019
Final corner detection	0.0058	0.0053	0.0053	0.0053
Count corners	0.018	0.016	0.016	0.016
Convex hull	0.041	0.038	0.039	0.038
Test for right angles				
Final rectangle hypothesis				
Median filter	0.082	0.041	0.025	0.015
Sobel	0.052	0.026	0.014	0.008

Table 14: Results for the Connection Machine on the Low-Level Portion

Intel iPSC-2

The Intel Scientific Computers iPSC-2 is a distributed memory multiprocessor that consists of up to 128 Intel 80386 microprocessors that are linked by a virtual cut-through routing network which simulates point-to-point communications. Each of the microprocessors can have up to 8 MB of local memory, and an 80387 arithmetic coprocessor. The benchmark implementation for the iPSC-2 was developed by the University of Illinois at Urbana-Champaign using C with a library that supports multiprocessing. The group had only enough time to implement the median filter and Sobel steps of the low-level depth image processing. However, they did run those portions on five different machine configurations, with 1, 2, 4, 8, and 16 processors, and on four of the five data sets. Table 15 presents their results, which are divided into user time and system time (including data and program load time, and output time).

Configuration	1		2		4		8		16	
	User	System	User	System	User	System	User	System	User	System
Median Filter										
Sample	176.47	0.00	87.93	11.52	43.46	11.23	22.27	3.1	11.14	3.82
Test	75.45	0.00	37.72	10.88	18.99	10.84	9.66	3.15	4.84	3.87
Test2	60.84	0.00	30.36	11.48	15.25	11.45	7.63	3.73	3.81	4.19
Test3	60.83	0.00	30.36	11.12	15.25	11.23	7.63	3.49	3.82	4.03

Sobel										
Sample	78.63	0.00	39.32	3.53	19.68	3.00	9.84	2.37	4.92	2.91
Test	80.82	0.00	40.42	3.47	20.25	2.89	10.15	2.43	5.10	2.82
Test2	80.82	0.00	40.42	1.46	20.25	1.99	10.15	1.87	5.10	2.50
Test3	78.63	0.00	39.31	2.62	19.68	2.51	9.84	2.17	4.92	2.69

Table 15: iPSC-2 Results for Median Filter and Sobel Steps

Comparative Performance Summary

As mentioned above, the direct comparison of raw timings is not especially useful. We have attempted to provide as much information about each benchmark implementation as is necessary for others to make informed and intelligent comparisons of the results. For example, a valid comparison of architectural features should take into account the technology, instruction rate, and scalability of the processors that were actually used to obtain the results. On the other hand, a comparison that seeks to establish the currently available machine with the best cost to performance ratio should look at the timings with respect to both the programming effort required and the price of the hardware. The authors hope to develop and publish some direct comparisons of architectural features, once a few more implementations are added to the sample and a reasonably broad set of scaling functions is established.

In the meantime, one interesting comparison that can be immediately drawn from the data, which requires no scaling for technology, is the relative amount of processing time that each architecture expends on each portion of the benchmark. This function, which is just the percentage of the total time taken for each step, provides an indication of those tasks that each architecture excels at and those that it struggles with. Tables 16 through 20 compare the efforts for the different architectures on each of the major benchmark steps, for the five data sets. It should be noted that the data sets Test and Test2 require very little model matching effort since they involve very simple models. The other three data sets involve more complex models, which is easily seen in Tables 16, 19, and 20. Only the complete implementations are listed, since a total time for the benchmark is required to compute the values in the tables. Blanks in the tables represent information that was missing from the reports by the different groups.

Architecture	Sun-3	Alliant	IUA	ASP	Sequent	Warp
Overhead	0.6	14.6	16.5	0.6	2.3	
Label connected components	3.5	12.0	0.1	30.0	4.9	9.1
Rectangles from intensity	0.8	5.8	19.1	2.5	0.6	
Median filter	30.9	16.8	0.7	0.6	11.9	24.5
Sobel	17.0	6.3	3.2	0.5	5.8	1.1
Initial graph match	3.1	4.3	14.4	0.0	1.2	1.0
Match data rectangles	0.0	0.2	3.5		0.0	0.5
Match links	0.0	0.1	10.5		0.1	0.5
Create probe list	0.0	0.0	0.1		0.0	
Partial match	3.0	4.0	4.0		1.1	
Match extension	40.9	34.2	35.6	63.9	61.9	56.9
Result presentation	3.1	5.4	2.7	0.7	8.0	6.0

Table 16: Distribution of Processing Time for Data Set Sample

Architecture	Sun-3	Alliant	IUA	ASP	Sequent	Warp
Overhead	1.5	26.4	30.9	2.3	3.1	
Label connected components	8.2	28.9	0.1	63.4	9.4	19.9
Rectangles from intensity	1.2	6.4	27.5	8.1	0.9	
Median filter	35.2	17.7	1.2	0.2	34.6	42.9
Sobel	39.4	11.4	5.9	1.7	34.4	2.4
Initial graph match	7.4	7.5	16.8	0.0	6.0	1.2
Match data rectangles	0.0	0.2	3.4		0.0	0.8
Match links	0.0	0.1	12.5		0.1	0.4
Create probe list	0.0	0.0	0.3		0.0	
Partial match	7.3	7.2	16.9		5.8	
Match extension	3.4	2.7	3.9	0.4	5.9	17.9
Result presentation	3.6	5.6	2.1	1.2	5.8	11.1

Table 17: Distribution of Processing Time for Data Set Test

Architecture	Sun-3	Alliant	IUA	ASP	Sequent	Warp
Overhead	1.7	26.5	30.6	2.0	3.2	
Label connected components	8.6	21.6	0.1	57.3	9.8	20.6
Rectangles from intensity	1.3	6.6	29.6	7.2	1.3	
Median filter	28.2	13.1	1.2	1.3	26.9	6.2
Sobel	41.3	11.5	5.9	1.6	34.8	3.2
Initial graph match	7.9	8.1	14.7	0.0	6.7	1.0
Match data rectangles	0.0	0.2	2.9		0.0	0.7
Match links	0.0	0.1	10.9		0.3	3.7
Create probe list	0.0	0.0	0.2		0.0	
Partial match	7.9	7.7	15.1		6.4	
Match extension	5.7	4.6	5.5	0.4	9.2	20.5
Result presentation	5.1	7.2	2.6	1.1	7.9	11.3

Table 18: Distribution of Processing Time for Data Set Test2

Architecture	Sun-3	Alliant	IUA	ASP	Sequent	Warp
Overhead	1.0	16.2	3.3	0.7	1.6	
Label connected components	5.1	13.4	0.0	30.8	4.9	7.8
Rectangles from intensity	1.0	5.8	3.1	2.8	0.7	
Median filter	16.5	8.0	0.1	0.5	13.2	2.4
Sobel	24.5	7.0	0.6	0.6	17.1	1.6
Initial graph match	12.4	14.1	26.9	0.0	8.1	2.1
Match data rectangles	0.1	0.4	3.2		0.1	0.7
Match links	0.1	0.5	23.6		0.1	1.4
Create probe list	0.0	0.0	0.2		0.0	
Partial match	12.2	13.1	43.7		58.3	
Match extension	36.8	30.9	21.5	0.2	50.9	66.4
Result presentation	2.7	4.0	0.7	0.4	3.5	3.9

Table 19: Distribution of Processing Time for Data Set Test3

Architecture	Sun-3	Alliant	IUA	ASP	Sequent	Warp
Overhead	1.0	16.3	3.5	0.7	1.6	
Label connected components	5.1	13.5	0.0	26.3	5.1	8.2
Rectangles from intensity	1.0	5.7	3.3	2.8	0.7	
Median filter	16.4	8.1	0.1	0.4	13.5	3.6
Sobel	24.5	7.1	0.7	0.5	17.5	1.7
Initial graph match	12.2	13.9	20.7	0.0	8.2	2.5
Match data rectangles	0.0	0.4	2.7		0.0	1.2
Match links	0.1	0.4	17.9		0.2	1.3
Create probe list	0.0	0.0	0.2		0.0	
Partial match	12.1	13.1	38.6		8.0	
Match extension	37.1	30.9	32.1	0.2	49.8	74.5
Result presentation	2.7	4.1	0.7	0.9	3.6	4.1

Table 20: Distribution of Processing Time for Data Set Test4

Recommendations for Future Benchmarks

At the conclusion of the Avon workshop, a panel session was held to discuss the benchmark, ways it could be improved, and future benchmark efforts. The general conclusion of the participants was that the benchmark is a significant improvement over past efforts, but that there is still work to be done.

One of the major complaints was the sheer size and complexity of the benchmark solution. The sample solutions are a considerable help in this regard, but a great deal of work is still required to transport them to parallel architectures. Several people expressed the opinion that a FORTRAN version should be made available so that the benchmark would be taken up by the traditional supercomputing community. It was pointed out that most groups don't have the time or resources to implement such a complex benchmark, and that it would be almost impossible to tune it for optimum performance as is done with smaller benchmarks. A counter-argument was voiced that most vision applications are not highly tuned, and that the benchmark might therefore give a more realistic indication of the performance that could be expected. Suggestions for reducing the size of the benchmark included removing one of the top-down probes (although there was no consensus on which one should be removed), and simplification of the graph matching code through increased generality.

On the other hand, several people complained that the benchmark task was too small. The groups that had benchmarked data-parallel systems all indicated that they would like to see data sets involving thousands of models so that they could exploit more data parallelism, rather than being forced into a task parallel model. Of course, those who had benchmarked multi-tasking systems took the opposite view. It was then suggested that an interesting variation on the benchmark would be to provide a range of data sets with model-bases ranging through several orders of magnitude. Such data sets would provide another dimension to the performance analysis, and thus some insight into the range of applications for which an architecture is appropriate. Beyond simply increasing the size of the model-base, several of the vision researchers expressed a desire to see a broader range of vision tasks in the benchmark. For example, motion analysis over a succession of frames would test an architecture's ability to deal with real-time image input and would help to identify those with a special ability to pipeline the stages of an interpretation. However, there was an immediate outcry from the implementors that the benchmark is already too complex. It was then suggested that an optional second level of the benchmark could be specified that would be based on the basic task, but extended to include image sequences and motion processing.

An important observation was made that the complexity of the benchmark was not the issue, but the cost of implementation. It was suggested that the benchmark might be more palatable if it was reorganized to be built out of a standard set of general purpose vision subroutines. Even though a group might still have to implement all of those routines, they would then at least have a library that could be used for other applications, over which they could amortize the cost. The benchmark specification would then be a framework for applying the library to solve a problem, and could involve separate tests for evaluating the performance and accuracy of the individual subroutines.

Part of the discussion focussed on the fact that the benchmark does not truly address high-level processing. However, as the benchmark designers were quick to point out, there is no consensus among the vision research community as to what constitutes high-level processing. Until agreement can be reached on what types of processing are essential at that level, it will be pointless to try to design a benchmark that includes the high level. It was also noted that the current top-down direction of low-level processing by the benchmark has some of the flavor of the high-level control of intermediate- and low-level processing which many people feel is necessary. In the end, it was decided that the community is not yet ready to define high-level processing to the degree necessary to build a benchmark around it.

Another point was that a standard reporting form should be developed, and that the sequential solution should output its results to match that form. Although the benchmark specification included a section on reporting requirements, the sequential solution did not precisely conform to it (partly because many of the reporting requirements were for aspects of the implementation that went beyond the timings and statistics that were to be output). In fact, most of the groups followed the example of the reporting format for the sequential solution, rather than what was requested in the specification. It was also noted that because the benchmark allows alternate methods to be used whenever dictated by architectural considerations, the reporting format can not be made completely rigid.

The conclusion of the panel session was to let the benchmark stand as specified for some period of time, in order to allow more groups to complete their implementations. Then a new version of the benchmark should be developed with the following features: It should be a reorganization of the current problem into a library of useful subroutines and an application framework. A set of individual problems should be developed to test each of the subroutines. A broader range of data sets should be provided, with the size of the model-base scaling over several orders of magnitude, and perhaps a set of images of different sizes. The graph matching code should be simplified and made more general purpose. A standard reporting format should be provided, with the sample solutions generating as much of the information as possible. Lastly a second level of the benchmark might be specified that extends the current problem to a sequence of images with motion analysis. The second level would be an optional exercise that could be built on top of the current problem to demonstrate specific real-time capabilities of certain architectures.

Conclusions

The DARPA Integrated Image Understanding Benchmark is another step in the direction of providing a standard exercise for testing and demonstrating the performance of parallel architectures on a vision-like task. While not perfect, it is a significant improvement over previous efforts in that it tests performance on a wide variety of operations within the unifying framework of an overall task. The benchmark also goes a long way toward eliminating programmer knowledge and cleverness as a factor in the performance results, while providing sufficient flexibility to allow implementors to take advantage of special architectural features.

Complete implementations have only been developed for a handful of architectures to date, but it is hoped that others will be added to the sample. In the meantime, it is possible to draw a few general conclusions from the data that has been gathered. It is clear that a tremendous speedup is possible for the data parallel portions of the interpretation task. However, every one of the architectures in this sample devoted the greatest percentage of its overall time to the model matching portion of the benchmark on those data sets that involved complex models. One conclusion might be that this portion of the task simply doesn't permit the exploitation of much parallelism. However, when the model matching step is viewed at an abstract level, it appears to be quite rich with potential parallelism, but in the form of task parallel direction of limited data parallel processing. While this style of processing can be sidestepped by increasing the size of the model-base so that the entire task becomes data parallel in nature, the inclusion of true high-level processing will force us back to dealing with this processing model. Thus, one potential area for research that the benchmark points out is the development of architectures, hardware and programming models to support task parallelism which can direct data parallel processing in a tightly coupled manner.

Acknowledgements

We would like to thank Claire Bono, Chris Brown, C.H. Chien, Larry Davis, Todd Kushner, Ram Nevatia, Keith Price, George Reynolds, Lew Tucker, and Jon Webb for their many helpful comments and suggestions in response to the draft benchmark specification.

For their efforts in designing, programming, and debugging the sequential and Sequent Symmetry solutions to the benchmark, we would like to thank Poornima Balasubramaniam, Sunit Bhalla, Chris Connoly, John Dolan, Martin Herbordt, Michael Scudder, Lance Williams, and especially Jim Burrill,

For providing benchmark results on other architectures we also thank Alok Choudhary, R. M. Lea, A Krikelis, I. Kossioris, Lew Brown, Dan Mezynski, Jon Webb, Lew Tucker, Steven Levitan, Mary Jane Irwin, Sunit Bhalla, Martin Herbordt, Michael Scudder, Michael Rudenko, and Jim Burrill.

And, for their many suggestions for improvements to the benchmark we again thank all of the above, plus Jake Aggrawal, Thomas Binford, Martin Fischler, Prasanna Kumar, Daryl Lawton, Randall Nelson, Karen Olin, Dennis Parkinson, Tomaso Poggio, Tony Reeves, Arnold Rosenberg, and Myung Sunwoo.

Bibliography

[Carpenter, 1987] Carpenter, Robert J., Performance Measurement Instrumentation for Multiprocessor Computers, Report NBSIR 87-3627, U.S. Department of Commerce, National Bureau of Standards, Institute for Computer Sciences and Technology, Gaithersburg, MD, August, 1987, 26pp.

[Duff, 1986] Duff, M.J.B., How Not to Benchmark Image Processors, in Evaluation of Multicomputers for Image Processing, L. Uhr, K. Preston, S. Levialdi, and M.J.B. Duff, Eds., Academic Press, Orlando, FL, 1986, pp. 3-12

[Hillis, 1986] Hillis, Daniel W., The Connection Machine, MIT Press, Cambridge, 1986

[Kung, 1984] Kung, H.T., and Onat Menzilcioglu, Warp: A Programmable Systolic Array Processor, Proc. SPIE Symp., Vol. 495, Real-Time Signal Processing VII, Aug. 1984

[Lea, 1988] Lea, R.M., ASP: A Cost-effective Parallel Microcomputer, IEEE Micro, October, 1988, pp. 10-29

[Preston, 1986] Preston, Kendall Jr., Benchmark Results: The Abingdon Cross, in Evaluation of Multicomputers for Image Processing, L. Uhr, K. Preston, S. Levialdi, and M.J.B. Duff, Eds., Academic Press, Orlando, FL, 1986, pp. 23-54

[Rosenfeld, 1987] Rosenfeld, Azriel R., A Report on the DARPA Image Understanding Architectures Workshop, Proceedings of the 1987 DARPA Image Understanding Workshop, February 1987, Los Angeles, CA, Morgan Kaufmann Publishers, Los Altos, CA, 1987, pp. 298-302

[Weems, 1988] Weems, Charles C., Allen Hanson, Edward Riseman, and Azriel Rosenfeld, An Integrated Image Understanding Benchmark: Recognition of a 2 1/2 D "Mobile", Proceedings of the 1988 DARPA Image Understanding Workshop, March, 1988, Cambridge, MA, Morgan Kaufmann Publishers, Los Altos, CA, 1988, pp.

[Weems, 1989] Weems, Charles C., Steven P. Levitan, Allen R. Hanson, Edward M Riseman, David B. Shu, and J. Gregory Nash, The Image Understanding Architecture, International Journal of Computer Vision, Vol. 2, Kluwer Academic Publishing, Boston, MA, 1989, pp. 251-282.

VLSI Processor Architectures for Computer Vision

Woodward Yang
Artificial Intelligence Laboratory, MIT, Cambridge, MA 02139

and

Alice M. Chiang
MIT Lincoln Laboratory, Lexington, MA 02173

Abstract

Efficient, real time execution of computer vision algorithms usually requires special purpose hardware. We present two types of processor architectures for performing localized operations on a single image and for performing shift and correlate type operations on multiple images. Efficient architecture designs should consider issues such as I/O bandwidth and practical VLSI limitations. CCD (charge-coupled device) technology has many desirable characteristics for computer vision processors. In particular, proposed CCD implementations of an edge detector, a reconstruction processor and a shift and correlate type processor will also be presented.

1. Introduction

Typical image processing tasks are very demanding for general purpose computers. Therefore, there is a need for special purpose processors especially for real time vision applications. The design of efficient image signal processors requires careful consideration of both the computational and I/O communication requirements. The goal of this paper is to present practical VLSI processor architectures that have particularly efficient CCD (charge-coupled device) implementations. This paper is organized as follows: First the processor architectures are presented. A review of CCD technology is followed by a presentation of the proposed CCD structures that implements an edge detector, a reconstruction processor, and a shift and correlate type processor.

2. VLSI Processor Architectures

Many vision algorithms consist of calculations or operations performed on spatially localized neighborhoods. Furthermore, most of the execution time is spent accessing pixel data values and not performing actual computations. Special purpose processors exploit the parallelism and regularity of the algorithm to achieve real time performance.

Parallel, Pipelined Architecture

In the design of a VLSI processor architecture, it is important to consider the degree of parallelism in processors and I/O connectivity, so that the computation speed and I/O capability are comparable. With an integrated **n** x **n** CCD imager, it is possible to provide **n** x **n** parallel processing elements, **n** parallel processing elements, or a single, serial processing element. While a fully parallel, **n** x **n** processor architecture potentially has the highest computational speed, it is typically I/O bandwidth limited since present VLSI technology cannot provide **n** x **n** parallel I/O paths. A single, serial processor is the simplest to implement. However, it typically requires large internal data storage capabilities and must operate at very high internal clock frequencies. A VLSI architecture using **n** parallel processing element with **n** parallel I/O paths provides a balance between computation speed and I/O bandwidth (see figure 1). By pipelining several stages of **n** parallel processing elements, a large variety of computations can be performed on a single image. As each column of pixels is serially shifted together, local interactions between neighboring column elements can be directly performed in parallel with each shift. Local interactions between neighboring row elements can be similarly performed by using time delay elements. Algorithms '.at can be decomposed into simple, localized operations can be efficiently realized in a parallel, pipelined architecture. This type of architecture is also particularly well suited for charge domain computations with CCDs. Furthermore, with an integrated CCD imager the n parallel processors can be placed outside of the imaging area so that the fill factor (amount of area devoted to photon collection) is not sacrificed.

Parallel, Pipelined Architecture

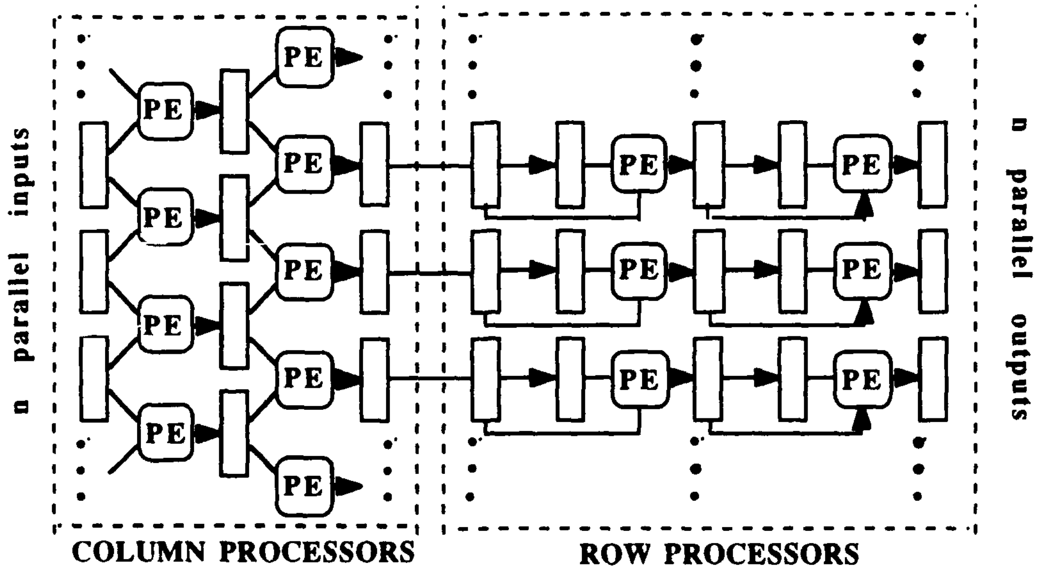


Figure 1. A schematic of the parallel, pipelined processor is shown. Rectangles represent pixel data values. Data is input and output in parallel as successive columns and is pipelined through the processor. The first block of local processing elements (PE) computes the interactions between neighboring column elements. The second block of local processing elements (PE) computes the interactions between neighboring row elements ty using delays.

Parallel, Shifted Architecture

A different type of processor architecture is required for algorithms that require multiple images. In particular, we consider motion and stereo algorithms that use local patchwise, shift and correlate operations. For motion analysis, the patchwise correlation of sequential images for a 2-D range of shifts is computed. For stereo, the patchwise correlation of left and right images is computed for only a 1-D range of shifts. Motion is seen to be the 2-D generalization of stereo. Data I/O is one of the most important constraints for processor architecture. If the image data are input as serial scan lines and the algorithm is to compute the correlation of s possible shifts, a processor architecture that utilizes s parallel processing elements as shown in figure 2 is able to efficiently manage data flow. Data in the processor flows smoothly to each processing element and does not need to by recycled after passing through the entire processor.

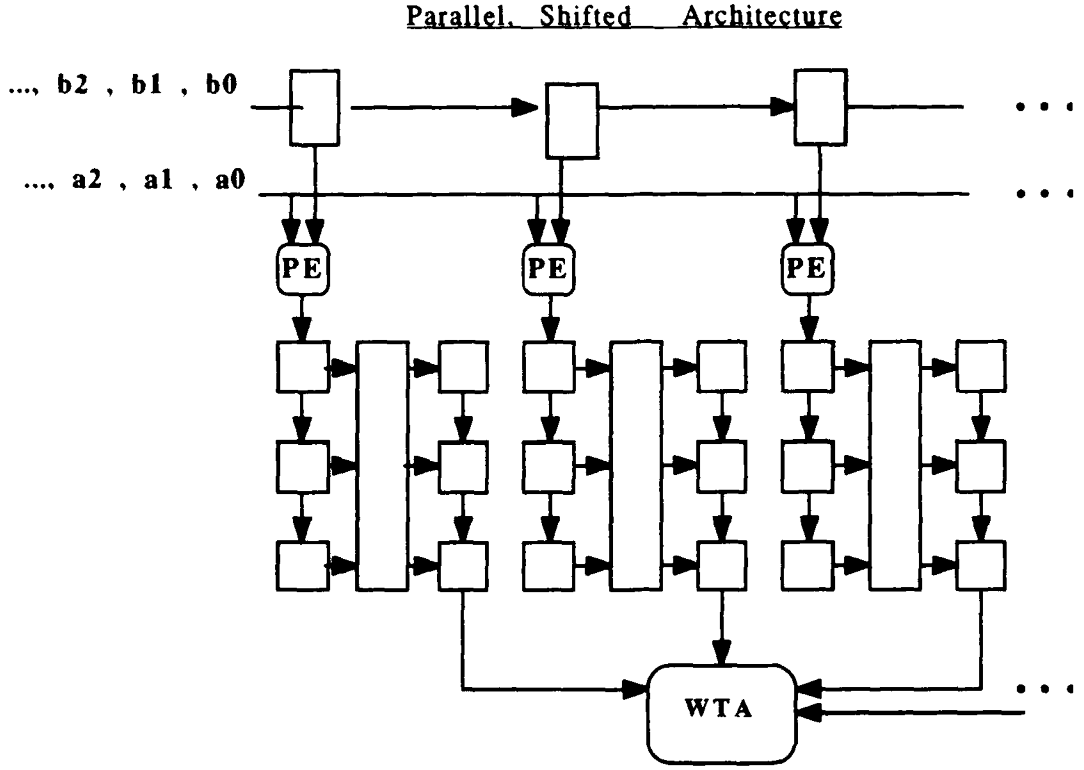


Figure 2. A schematic of the parallel, shifted architecture for shift and correlate type computations is shown. Two images ***a*** *and* ***b*** *are input as serial scan lines. Pixel values of image* ***a*** *are directly input to each processing element (PE). Pixel values of image* ***b*** *are input through a shift register so that each PE has a different pixel value of image* ***b*****.** *The* ***s*** *PEs (each corresponding to one of the range of s possible shifts) compute in parallel the correlation of a pair of pixels form image* ***a*** *and* ***b*****.** *As the next pixel of image* ***b*** *enters the shift register, the next pixel of image* ***a*** *is also input. Notice that each of the* ***s*** *PEs corresponds to a constant shift value. The computed correlations are passed into a SIPO (serial-in parallel-out) register. The SIPO register passes the computed correlations into a processor such as the binomial convolver that computes the average correlation over a neighborhood. The patchwise correlation of a pixel for a given shift is serially output by a PISO (parallel-in serial-out) register. The correlations are compared by a non-maximum suppression (WTA) operation and the shift value (motion vector or tereo depth) of maximum correlation is output at the same pixel serial input rate.*

3. CCDs

CCDs operate as shift registers that store information as an analog value of charge. A CCD shift register is essentially an analog, sampled data memory. Charge can be stored for ~30 milliseconds and clocked at ~100 MHz for typical silicon CCDs. The actual bit accuracy or dynamic range is determined by the maximum and minimum charge values which are a function of the actual size of the device, physical noise, and sensitivity of the output charge-to-voltage converter. Using typical process parameters for silicon CCDs, 8-bit accuracy is readily achievable. A single CCD is capable of serving as memory by storing an analog charge value as well as performing simple operations such as addition, subtraction, multiplication by a constant, division by a constant, delay, and data I/O. CCDs are particularly well suited for pipelined or parallel architectures since they are analog shift registers and are easy to configure and clock in parallel. In addition, CCDs are used extensively as imaging devices. Image signal processing and computer vision are ideal applications for CCD technology since the CCD signal processors and CCD imager can be integrated together which potentially increases the I/O bandwidth between the imager and signal processors. In the past, CCDs have been effectively used for high performance signal processing [1] as well as combined imaging and signal processing [2].

4. CCD Implementations

At present there are many successful algorithms implemented on general purpose computers that can perform early vision tasks, but there are few that are capable of real time performance. Possible implementations of early vision tasks using the parallel, pipelined architecture and the parallel, shifted architecture are presented below. By matching the algorithms with the capabilities of a CCD image processor, the proposed implementations of the parallel, pipelined and parallel, shifted architecture are capable of operating in real time and at the image frame rate.

Edge Detector

Edge detection is one of the most important early vision task. Edge positions contains much of the essential cognitive information of the scene. While there are many edge detection algorithms (i.e. Sobel operators, Canny edge detector, difference of Gaussians, etc.), the LoG (Laplacian of Gaussian) level-crossings was chosen for its ease of implementation. The CCD implementation of this edge detection algorithm is separated into two components, the 2-D Gaussian filter and the Laplacian operator. Once the LoG of the image intensity has been computed, a simple comparator with an adjustable threshold can be used to identify the level-crossings which correspond to edges in the image.

The CCD implementation of a 2-D Gaussian filter utilizes the fact that a 2-D Gaussian filter is decomposable into a 1-D Gaussian filter in columns and a 1-D Gaussian filter in rows. Furthermore, the CCD structure actually implements a binomial distribution which is a good approximation to a Gaussian. (The precise shape of the low pass filter is not critical since it is used primarily to improve the robustness of the algorithm to noise.) By combining CCD structures that perform the charge domain operations of division by 2 and addition, a simple first order 1-D binomial convolution within column elements is implemented in the charge domain by the CCD structure shown in figure 3a. The full 2-D Gaussian could be accomplished by transposing the image and using the same processors to compute the 1-D binomial convolution within row elements. However, transposing an image in the charge domain requires a full frame delay and complex hardware. A faster, more direct method of achieving a first order, 1-D binomial convolution within row elements is to use the same division by 2 and addition operators in conjunction with a delay element as in figure 3b. The order of the binomial distribution which determines size and extent of the approximated Gaussian filter is increased by successive applications of the first order binomial convolutions.

Basic CCD Structure for 1-D Binomial Convolution within Columns and within Rows

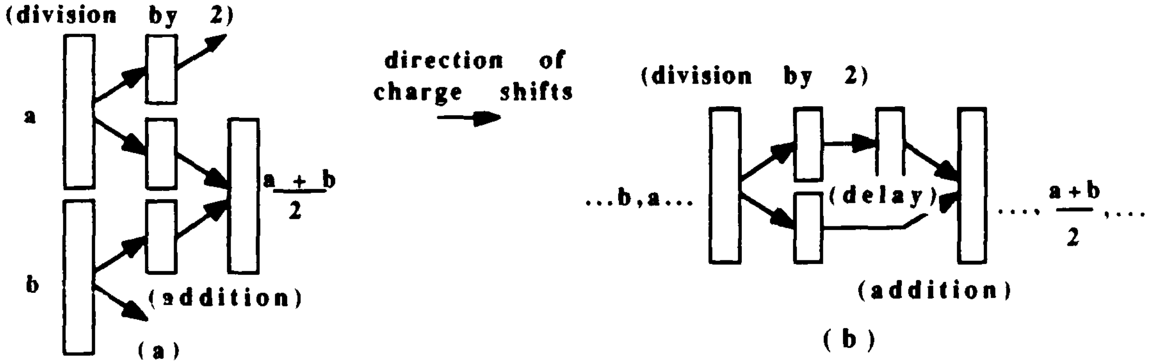


Figure 3 Basic CCD structure for computing binomial convolution. Repeated operations increase the order of the binomial and extent of weighted average. (a) As a column of pixel values is passed in parallel through the CCD structure, the signal charge is divided by 2 and summed to form the average of neighboring column elements. (b) As a row of data values is pipelined through the CCD structure, the signal charge is divided by 2, delayed, and summed to form the average of neighboring row elements.

The implementation of the Laplacian operator is simply a slight variation of the 2-D Gaussian filter. The Laplacian operator is implemented by a convolution with a 1/8 [0 1 0] , [1 -4 1], [0 1 0] mask. The CCD implementation computes the convolution of the positive values separately and subsequently subtracts the negative value to realize the Laplacian operator. The basic CCD structure that performs this operation is realized by combining the same simple charge domain operations of addition, division by 4, and delay with a subtraction operation and is schematically described in figure 4.

CCD Structure for Laplacian Computation

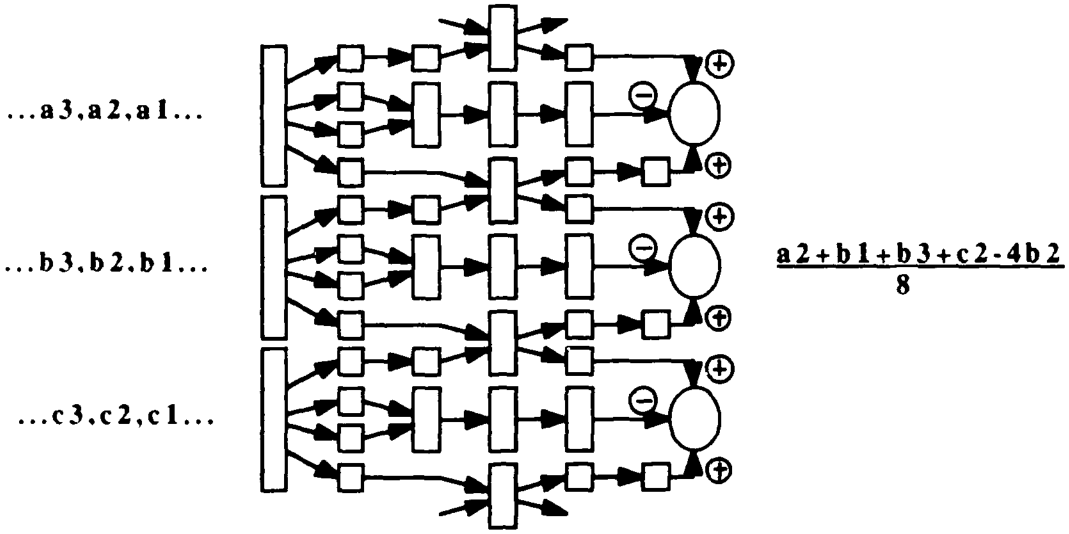
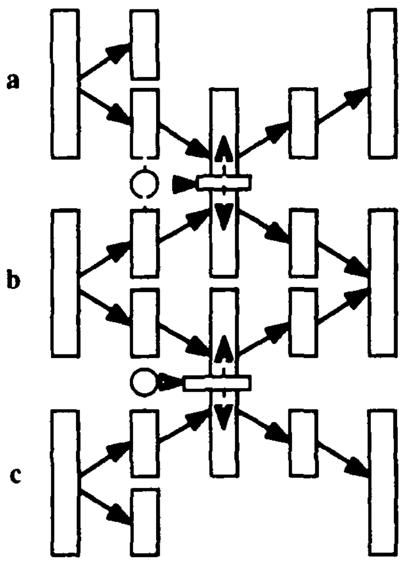


Figure 4. A CCD structure that computes the Laplacian is shown. As columns of data values are pipelined through, the charge is divided in 4, delayed, summed, divided by 2, and finally subtracted in order to implement a convolution with a 1/8 [0 1 0], [1 -4 1], [0 1 0] mask.

Reconstruction/Integration Processor

Gaussian convolution and other low pass filters are used to suppress high frequency components. However, linear low pass filters also reduce the high frequency components of real information in the image typically resulting in blurry object edges. Various nonlinear operations such as median filtering have been used to try to reconstruct or enhance images. In particular, a variation of a surface reconstruction algorithm used by Blake and Zisserman [3] is well suited for the parallel, pipelined architecture. The algorithm is essentially a conditional convolution that preserves large differences and filters small differences. In other words, a region is low pass filtered if the differences in intensity between neighboring pixels are below some threshold. This algorithm has been analyzed as a deterministic approximation of Markov random fields by Geiger and Girosi [4]. By combining a thresholding element that measures local differences with the binomial convolver, the reconstruction processor is implemented as shown in figure 5. Implementation of an independently adjustable thresholding element introduces the ability to integrate information from different image fields such as intensity, color, depth, motion, etc. In a real image, different fields are highly correlated (depth discontinuities tend to occur in conjunction with intensity edges) and should influence the reconstruction of other image fields. This reconstruction/integration processor is potentially a powerful application for CCD image processors.

Basic CCD Structure for Reconstruction Computation



b , $\frac{a+3b}{4}$,

$\frac{3b+c}{4}$, or

$\frac{a+2b+c}{4}$

Figure 5. A CCD structure that implements a surface reconstruction algorithm is shown. The circle and dashed lines represent a thresholding element that will prevent charge from being summed if the difference in data values exceeds a given threshold. As a column of data values is passed in parallel through the processor, the charge is split and is conditionally summed. The resulting output is some weighted average of neighboring column elements dependent on local differences. An integration processor is implemented by using an independently adjustable thresholding element.

Stereo/Motion Processor

CCD implementation of a shift and correlate type of operation for stereo or motion utilizes the parallel, shifted architecture. The processing elements need to compute the norm of the two input pixel values. An absolute difference operator is usually an acceptable norm for shift and correlate algorithms. Standard CCD input structures implement a subtraction operation when the first operand is less than the second operand and give a zero result when the first operand is less than the second. The absolute difference operator is implemented by summing the values from a standard CCD input structure and an identical CCD input structure except with the first and second inputs reversed.

5. Conclusion

A parallel, pipelined architecture and a parallel, shifted architecture for VLSI hardware implementation of vision algorithms was presented. The parallel, pipelined architecture is particularly well suited to image processing algorithms that can be decomposed into simple, localized operations. Possible CCD implementations of an edge detector and reconstruction/integration processor were described. The parallel, shifted architecture is well suited to shift and correlate type algorithms such as those for stereo and motion. A CCD implementation of the correlation operator as an absolute differencing processor was proposed. Data I/O is one of the most important constraints on a VLSI processor architecture. The ability of CCDs to serve as memory, I/O data path, and processor makes it a very promising technology for image processing and computer vision applications.

Acknowledgements

The authors are grateful to Tomaso Poggio for his guidance and many helpful discussions. Woodward Yang is supported by a fellowship from the Hertz Foundation.

References

[1] A. M. Chiang, "A New CCD Parallel Processing Architecture," in *Proceedings of CMU Conference on VLSI Systems and Computations*,, Pittsburgh, PA., pp. 408- 415, edited by H.T. Kung, B. Sproull, and G. Steele, Computer Science Press, 1981.

[2] J.P. Sage and A. Lattes "A High-Speed Analog Two-Dimensional Gaussian Image Convolver," *Technical Digest of the Optical Society of America Topical Meeting on Machine Vision*, Incline Village, NV, p. FD5-1, March 1985.

[3] A. Blake and A. Zisserman, Visual Reconstruction, MIT Press, Cambridge, MA., 1987.

[4] D. Geiger and F. Girosi, "Mean field theory for surface reconstruction," this proceedings.

GAZE CONTROLS WITH INTERACTIONS AND DELAYS

Christopher Brown
Computer Science Department
University of Rochester
Rochester, NY 14627

ABSTRACT

Five control systems loosely corresponding to primate saccadic, vergence, pursuit, vestibulo-ocular, and head control operate on a simulated two-eyed robot head maneuvered by a robot arm. The goal is to get some qualitative understanding of the interaction of such reflexes under various assumptions. The simulation is meant to be relevant to U. Rochester's robot. Thus it incorporates kinematics of the robot head but assumes a "tool-coordinate" system available to robot arm commands, so that arm kinematic calculations are unnecessary. Dynamics are not modeled, since they are handled by the commercial controllers currently used in the Rochester robot. Even small delays render the effect of delay-free controllers unstable, but multi-delay version of a Smith predictor can to cope with delays. If each controller acts on the predicted system and ignores other controllers, the situation is improved but still potentially unstable if controllers with different delays act on the same control output. The system's performance is much improved if controllers consider the effect of other controllers, and the resulting system is stable in the presence of a certain amount of stochastic disturbance of control delays and inputs, and also in the presence of systematic error arising from inaccurate plant and world models.

INTRODUCTION

Behaving, actively intelligent (mechanical or biological) systems must manage their computational and physical resources in appropriate ways in order to survive and to accomplish tasks. At Rochester we are building an integrated actively intelligent system that incorporates abstract reasoning (planning), sensing, and acting [Bro88]. The *active intelligence* paradigm we shall exploit incorporates the following ideas.

1. A hierarchy of control, so that the highest cognitive levels can reason in terms of *what* they want done rather than *how* to do it in detail. This hierarchy should extend throughout the system.
2. At the lower levels, the control hierarchy ends with visual and motor skills or *reflexes*. These capabilities are cooperative but to some extent independently controllable. Some are always running, and they form the building blocks on which more complex behavior is built. Examples are tracking targets to minimize motion blur or redirecting gaze as a result of attentional shifts.
3. Part of the job of low-level visual capabilities is to present perceptual data, such as flow fields or depth maps, to higher-level visual processes. Low-level processes can often benefit from knowledge of self-initiated motion on the part of the sensing entity. They can often be built on the low-level control capabilities.

We currently have a nine degree of freedom robot body-head combination controlled by a Sun computer interfaced over a serial line to a VAL-II robot control system, and over a VME bus to the three eye motor controllers. The visual input is processed by a pipelined image processing system. The system has been used in several promising demonstrations of considerable complexity in depth-map creation and vergence

([BO88,OP89]). It has also been used for some simple but effective real-time applications in tracking and fixation.

What has been missing so far has been the cooperation of several modes of control, or the operation of several at once. In the work reported below, a simulation of the robot head and eyes is used to examine the effects of different styles of interacrion between certain control capabilities that we have implemented (such as tracking) or anticipate using (such as using eye movements to compensate for head movements).

The simulation software is based on the actual robot head kinematics, and has provided a flexible tool for investigating the interaction of different control methods and different types of control interaction.

THE MODEL OF HEAD AND IMAGING

The simulator geometry can capture all the essentials of the Rochester robot [Bro88,BR88] (including the annoying "non-spherical" geometry of the camera pans and tilts). It allows geometric parameters to be changed to explore the effects on error and the possibility of adaptative control. The robot arm is not modeled: rather the model abstracts it to a single eye-support platform that can be postioned arbitrariiy in space with six degrees of freedom: three in position, three in orientation. On the model head is a modelled tilt capability that affects both cameras, and each camera has a modelled pan capability. The geometry of the offsets of the various axes in these links are variable, and incorporate the geometrical complexity of the real system. The simulated mechanism is massless; this reflects the effective behavior of our current hardware system when viewed from its high-level control operations. The independent control of the camera pans allows us to model modern theories of saccadic and vergence systems; heads with mechanical vergence capability need one fewer motor but must use older models of these systems.

The camera models incorporate point projection with fixed focal length, as well as a "foveal-peripheral" distinction by which the location of imaged points is less certain, outside a small foveal region, depending on the off-axis angle of the target being imaged. The target itself is a single point in 3-D space, moving under dynamical laws. The experiments below were often carried out with the target point in orbit about an invisible "black hole" - thus the target followed an elliptical path. In other experiments the target moved in a straight line. In some of the experiments involving delays the target was stationary but the robot moved in X, Y, and Z, thus creating a perceived target motion, but one due to factors under robot control.

It is assumed that the imaging system knows the distance to the target (in real life, this distance may be derived from binocular stereo, apriori knowledge, any of a number of monocular distance cues, kinetic depth calculations, etc.). It is assumed that, for each eye, the instantaneous retinal velocity of the target is known (i.e. the vector difference between its position in the current image and its position in the last image). Other than that, the system only knows the left and right image (x,y) location of the target's image. Of course the target's image position and hence image velocity is perturbed by uncertainties arising from the blurriness of peripheral vision, should the target not be foveated. There is a further provision to add uniform noise to the target's imaged position - this can model quantization noise, or be used to approximate process noise in the target's motion.

THE MODEL OF CONTROL

ZERO DELAY CONTROL

The input to the control systems is usually based on quantities that can be inferred from vision (e.g. the (x,y) position of the target, which should be driven to (0,0), or target disparity between the two eyes which should be driven to 0). Some control inputs arise from the robot's "proprioception" (e.g. the amount the cameras are panned or tilted from their null position), and some is from other control signals (when one control is to null out the effects of another). The simulation has controllable output parameters corresponding to one set of VAL-II robot control parameters (the VAL-II "tool coordinate system") for the head: its X,Y,Z position and A,B,C orientation. Also there is direct control over the pans (independent for left and right) and tilt (common) of the two cameras. In every case the outputs of controls are velocity commands to the nine degrees of freedom in the system, reflecting one simple form of our current interface to the motor controllers.

The basic control loops that manage the system are loosely inspired by the primate visual system. However, most assumptions and technical decisions have been made either for the sake of simplicity or to mimic our robot rather than for the sake of faithfully modelling known biological systems or optimal mechanical systems (see the Discussion section below). Still, one of the major design goals is that the system can support more detailed control models. Most of the loops have several parameters, such as the proportional, integral, and derivative (PID) constants of their controllers, and their delays and latencies. Delay means the amount of time after a commanded motion before it commences — this is often called latency in the literature. Latency is how long it takes the command to complete: it is another time constant that indicates both how soon another command can be accepted, or how long the command will be affecting the controlled (velocity) variables. In all the work so far, only saccades have latency greater than unity. In the robot system the delay correponds to how long it takes the mechanical system to respond to a motion ordered from a high software level, and the latency reflects how long it takes to complete a command. The assumption is of control delay, not sensor delay: that is, we assume that "sensors" (visual or robot- and eye-control motor states read from their controllers) are available to the system immediately, without delay, and thus reflect the true state of the world. (Our analysis and the algorithms extend to the case that the sum of control and sensor delays is constant for any controller.)

There are five separate control systems.

1. Saccade: fast slewing of cameras to point in commanded direction. Saccades are modelled as open loop, though in primates there are "secondary" saccades that correct errors in initial saccades. The saccadic system tries to foveate the target and to match eye rotations to the target velocity so as to be tracking the target as soon as the saccade is completed. Current opinion is that the saccadic system is aware of the 3-D location of the target, not just the location of its retinal image. However, in the implementation used for the experiments below, saccades operate with retinal locations and velocities, not 3-D locations or distance. The left eye is dominant in the system. The saccade aims to center the target image on the fovea of the left eye; the right eye is panned by the same amount (and of course tilted by the same amount for mechanical reasons). Thus the saccade maintains the current vergence angle. It is implemented as a constant-speed slewing of all three pan and tilt axes, with one of them attaining a system constant maximum velocity. The slewing continues until the target should be foveated (it my not be due to peripheral blurring or other noise), at which time the system is left with eye velocities that match the perceived target motion before the saccade. The saccadic system is characterized by its maximum velocity and its delay.

2. Smooth Pursuit: tracking a moving target. This is a "continuous" activity as opposed to the discontinuous saccadic control activity. The error here is target position in the left eye, (which should be (0,0)), and the commands are pan and tilt velocities to the left eye. The pursuit system has delay, latency, and PID control. In both the saccadic and smooth pursuit systems modeled here, there is strict (exclusive) left-eye dominance.

3. Vergence: the vergence system measures horizontal disparity between the target position in the left and right eyes, and pans the right eye to reduce it. The vergence system has delay, latency, and PID control.

4. Vestibulo-Ocular System: the VOR system is open loop in the sense that its inputs come from the head positioning system and its outputs go to the eye positioning system. Its purpose is to stabilize

eyes against head motion, and its inputs are the control signals for head position (XYZ velocities, ABC angular velocities). It also uses the distance of the target, since that affects the appropriate response. The VOR should ideally be implemented by inverse kinematics, to which the current implementation (and presumably the neural one) is an approximation. Its output is commands to the pans and tilt controls to null out the apparent target motion caused by head motion. It is characterized by delay, latency, and open loop proportional gain.

5. Platform Compensation: This system is a head-control, not gaze-control system. These systems are known to interact in subtle and complex ways, but this particular reflex simply attempts to keep the eyes "centered in the head", so that the camera pans or tilts are kept within "comfortable" mechanical ranges. The "*comfort function*" is a nonlinear one $x/((x - xmax)^2)$, where x is the average pan angle (to control head "yaw" movements) or the tilt angle (to control head "pitch" movements). In either case $xmax$ is the mechanically imposed limit of the system. This reflex is open loop (eye position affects head position), with delay, latency, and open loop proportional gain.

The system has the capability of operating in two modes: ***smooth pursuit*** and ***saccade***. In smooth pursuit mode, the VOR, platform compensation, pursuit, and vergence systems are left running. In saccade mode, other controls may be diabled. This allows modelling the effects of turning off vergence, head compensation, tracking, *etc.* during saccades. Ultimately it seemed best only to turn off tracking during saccades, but other combinations are demonstrated below.

The delays and latencies are implemented with a command pipeline, in which the commanded changes in velocities are entered opposite the time in the future they are to take effect. Time is discretized to some level, called a *tick* henceforth. A larger delay results in entry of the corresponding command further in the future. Latencies are implemented by dividing the commanded change between as many discrete time periods as necessary to spread the effect over the latency. The pipeline thus is indexed by (future) time instant, and it has entries that hold the commanded velocities for the six head degrees of freedom and three camera degrees of freedom. Each instant also has an entry corresponding to its mode (saccadic or pursuit). The pipeline is implemented as a ring buffer.

For the delay-free case, the control architecture is strictly independent. That is, controllers are ignorant of each other's effects, and the combination of control effects is modeled by all controllers incrementing or decrementing a common control register (indicating some motor velocity setting). All increments and decrements are made to the current value that is there already, which perhaps is nonzero because of input from another reflex. Thus the control commands are summed in the simplest possible way, as if each control system's output were a D.C. voltage and all the outputs were soldered together at the effector motor's input.

The saccadic system shuts down the pursuit system in the sense that for the duration of the saccade (which is computed from the image distance it must move the fovea and the maximum velocity it can move), all other commands in the pipeline are overwritten, and the mode is changed to "saccade". Further commands trying to affect these instants may be ignored, depending on the (compile-time) policy desired.

NON-ZERO DELAY CONTROL

Slight amounts of delay destabilized the simulated system, as expected (see the Experiments section below). Control with delays can be stabilized by turning down gains and slowing the response of the system, but its performance then suffers. Successful control with delays incorporates some form of prediction [Mar79]. The controller implemented in the simulation is a version of a Smith predictor [Smi57,Smi58], which is the basic idea behind most modern methods.

Smith's Principle is that the desired output from a controlled system with delay p is the same as that desired from the delay-free system, only delayed by the delay p. Let the delay be z^{-p}, the delay-free series controller be $C(z)$, the desired delay controller be $\tilde{C}(z)$ and the plant be $A(z)$. The delay-free system transfer function will be

$$\frac{CA}{1+CA}.$$

The delay system with its desired controller has transfer function

$$\frac{\tilde{C}Az^{-p}}{1+\tilde{C}Az^{-p}}.$$

But Smith's Principle is

$$\frac{\tilde{C}Az^{-p}}{1+\tilde{C}Az^{-p}} = \frac{CAz^{-p}}{1+CA}.$$

This quickly leads to the specification for the controller $\tilde{C}$ in terms of C, A, and z^{-p}:

$$\tilde{C} = \frac{C}{1+CA(1-z^{-p})}.$$

This simple principle has spawned a number of related controllers, often arising from each other by simple block-diagram manipulation. Figure 1 is one block diagram of a Smith prediction controller, and it describes the implemented system in the simulator.

If the maximum delay of a controller in the system is T, The plant model is a pipeline of enough future robot states to reach time T into the future, updated and extended once a tick. Ideally the robot's state is predictable, since only the control commands act on it. Practically there may be some plant noise. In the work so far, the world prediction is simplified by assuming the world is static and that the robot does all the moving (navigation in a static environment). As part of the experiments, target motion was added to test the system's response to a false target model.

EXPERIMENTS

DELAY-FREE CONTROL

In all the simulations, the goal of the system is to put one or both of its eyes squarely on the target (at retinal position (0,0)) and keep them there. The head is always in an upright position, so pans rotate the cameras about a vertical world axis, tilts rotate the cameras about a horizontal axis. With a static head, pans induce image x motion upon a static, foveated target and tilts induce image y motion. In all the graphs of this section, the horizontal axis is time, and the vertical axis is pan and tilt error, or equivalently the image x and y position of the target. Each graph shows both left and right eye x and y errors, but often the y errors are superimposed since the tilt platform is common to both cameras. In every case there is "peripheral blur", which is modelled by adding, outside a small "fovea", uniform noise to the target (x, y) location, with standard deviation proportional to $1/d$, where d is the euclidean distance of (x, y) from the $(0, 0)$ point. The simulation does not use realistic time-constants and speeds, which instead are scaled so that interesting effects happen within a few ticks.

Figs. 2 and 3 illustrate the cumulative effect of simply superimposing control capabilities: each operates independently and their outputs are simply summed at the effectors. Delays are zero, latencies (except for saccades) unity. In these two figures tracking is by position error signal.

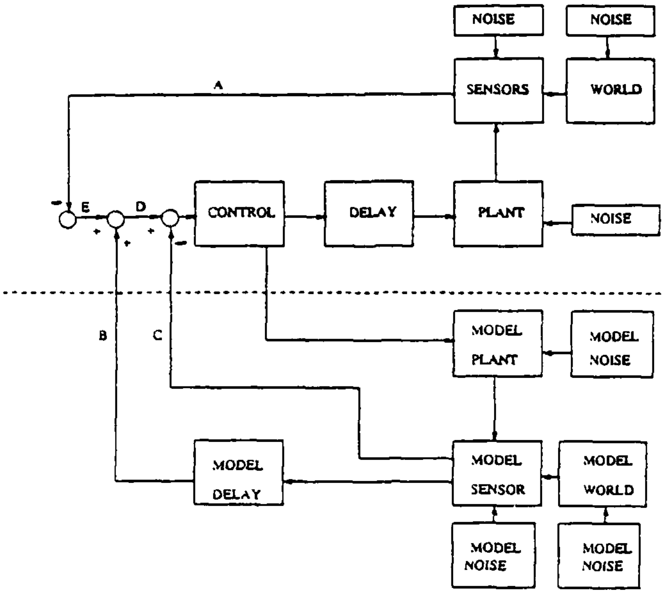


Figure 1: The implemented Smith predictor control. The block diagram is easily derived from the Smith predictor equation, with the MODEL PLANT, MODEL WORLD, and MODEL SENSOR blocks corresponding to A. $\tilde{C}$ is represented by the block labelled CONTROL and everything below the dashed line. The CONTROL block represents all five control systems, and the DELAY block represents a vector of their five independent delays. The PLANT, WORLD, and SENSOR blocks represent the robot simulation. Delayed control is implemented with a pipeline of controls to take place in the future, and the plant model is a similar pipeline of predicted robot states derived from the control.

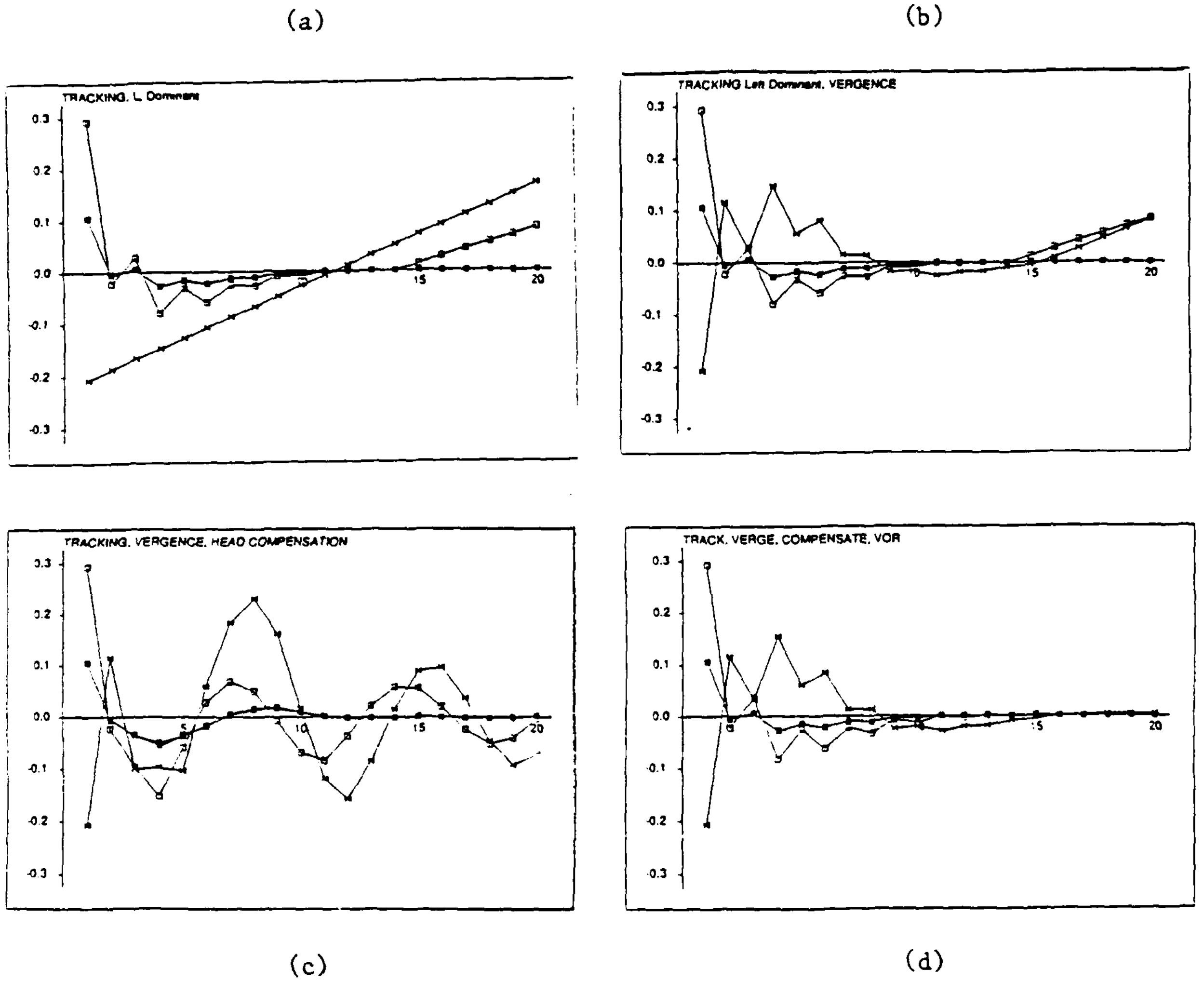
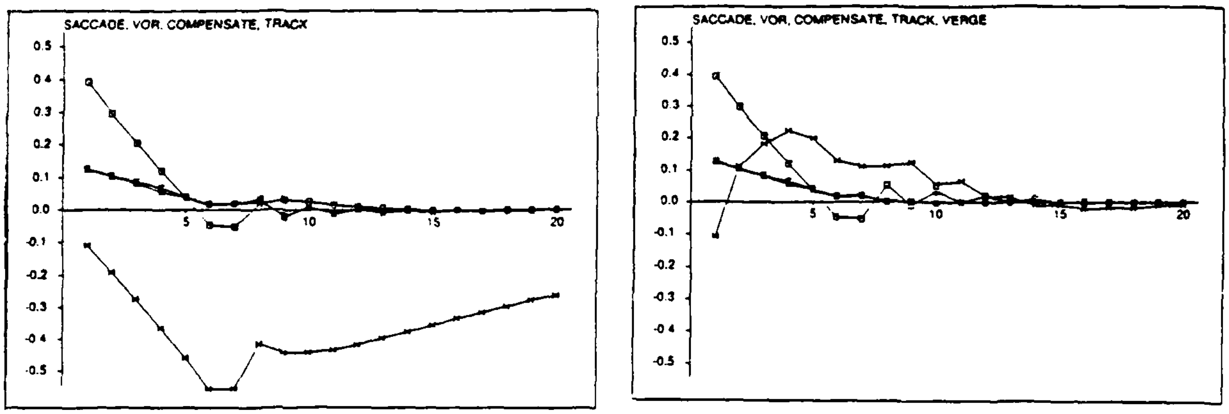


Figure 2: Increasingly effective delay-free control results from superposition of noninteracting controllers. (a) Tracking only: The left (dominant) eye pans and tilts, inducing tilt in the right eye. The tracker uses a position error signal. The right eye gets no pan signal, and its horizontal error accrues from target motion. The left eye tracks successfully until it hits mechanical stop at tick 14. (b) Add vergence: Both eyes hit stops at about tick 15. (c) Add head compensation: This control is to keep eyes from hitting mechanical stops by turning the head in the same direction as the tracking motion. A less-desirable effect is to amplify the tracking signal, overcompensating and destabilizing the tracking. (d) Add VOR, which effectively compensates the head rotation with eye rotations.



(a) (b)

Figure 3: (a) Continuing the previous figure with tracking driven by position error, add saccades in which vergence, VOR, and head compensation are turned off during saccade. The saccade drives the left eye error more or less to zero (it is affected by the peripheral blurring effect which makes the initial location of the target image uncertain). It slews the right eye off target. When VOR, head compensation and vergence are turned on after the saccade the first two reflexes have a transient effect. (b) Here let vergence run during the saccade but inhibit VOR and head compensation until after saccade completes.

Fig. 4 shows the effects of tracking with a velocity error signal. Here saccades are initiated if the target falls outside a fixed distance (here .1) from the fovea.

Finally, Fig. 5 shows the effects of control delay on the system. The smallest delays, applied uniformly or to just one control, destabilize the system seriously.

DELAY CONTROLS

As derived, the Smith predictor is appropriate for a single system control (or sensing) delay. In our system there will be differing delays reflecting different software actions (serial line plus VAL-II software versus VME-bus connection to the eye motor controllers, for instance). The idea of the Smith predictor is easily extended, however.

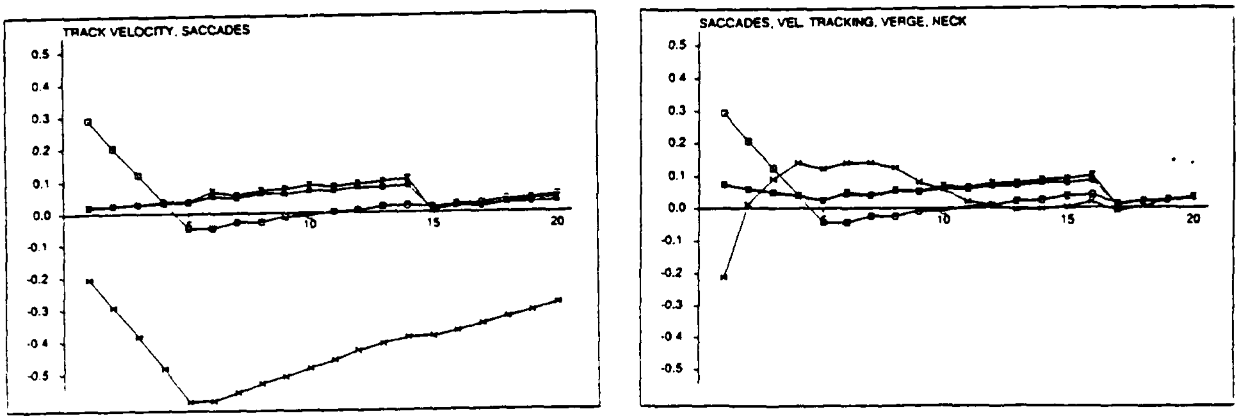
Independent Delay Control

Two types of control were implemented using the Smith controller of Fig. 1. In the first, the controllers are ignorant of the delays of other controllers, and also ignorant of the sharing of output variables between controllers. Each controller knows its own delay T, and uses the following algorithm. *Look ahead time T and retrieve the predicted robot and control states for that time. Apply the control appropriate for these future states now.*

Fig 6 shows some sample effects of this independent delay-control strategy. The system is stable for certain combinations of delays, but is unstable unless all the non-vergence delays are the same.

Interacting Delay Control and Noise

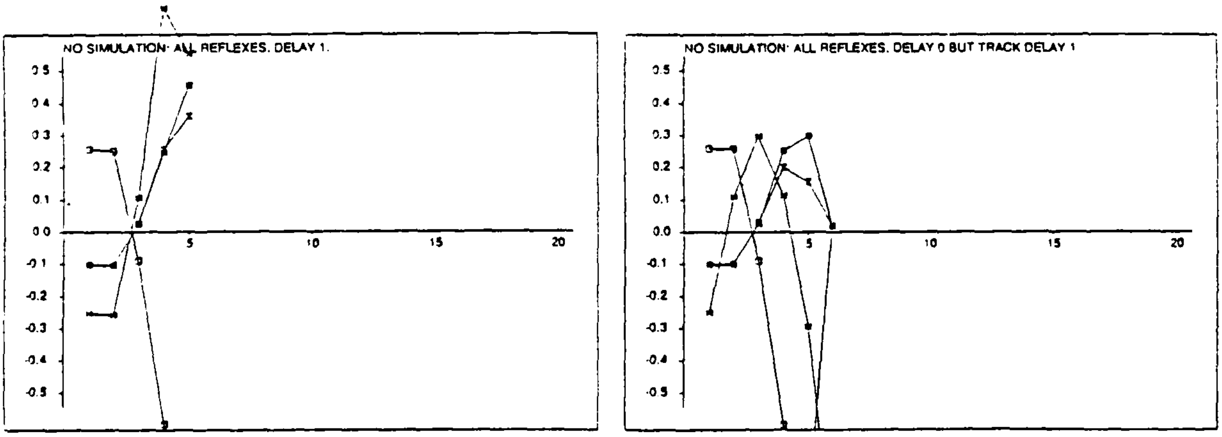
The independent delay control algorithm is not as smart as it could be. The short-delay controls do not look into the future as far as the long-delay controls, and therefore they do not anticipate the effects of slower controls. This effect shows up when long-delay and short-delay controls affect each other's output, either directly or through the kinematic chain. The reason the verge reflex can run with different delay and not destabilize the independent delay control system is that no other control (barring saccade) affects the right camera's pan velocity, and panning is at the end of the kinematic chain. Assume each controller knows its



(a)

(b)

Figure 4: (a) No vergence, velocity-error tracking with saccades for position control. Tracking is subject to steady-state position error. (b) Add vergence, and also change head kinematics (unknown to any controllers) from a "spherical" geometry to the Rochester robot's configuration of pan, tilt, and optic axes. The changed geometry has little effect.



(a)

(b)

Figure 5: (a) The no-delay controller applied to the system with a constant delay of one tick in all controls. Ideally this graph should be a delayed version of Fig. 2(d). (b) The no-delay controller applied with zero delay in all controls except tracking, which has a delay of one tick.

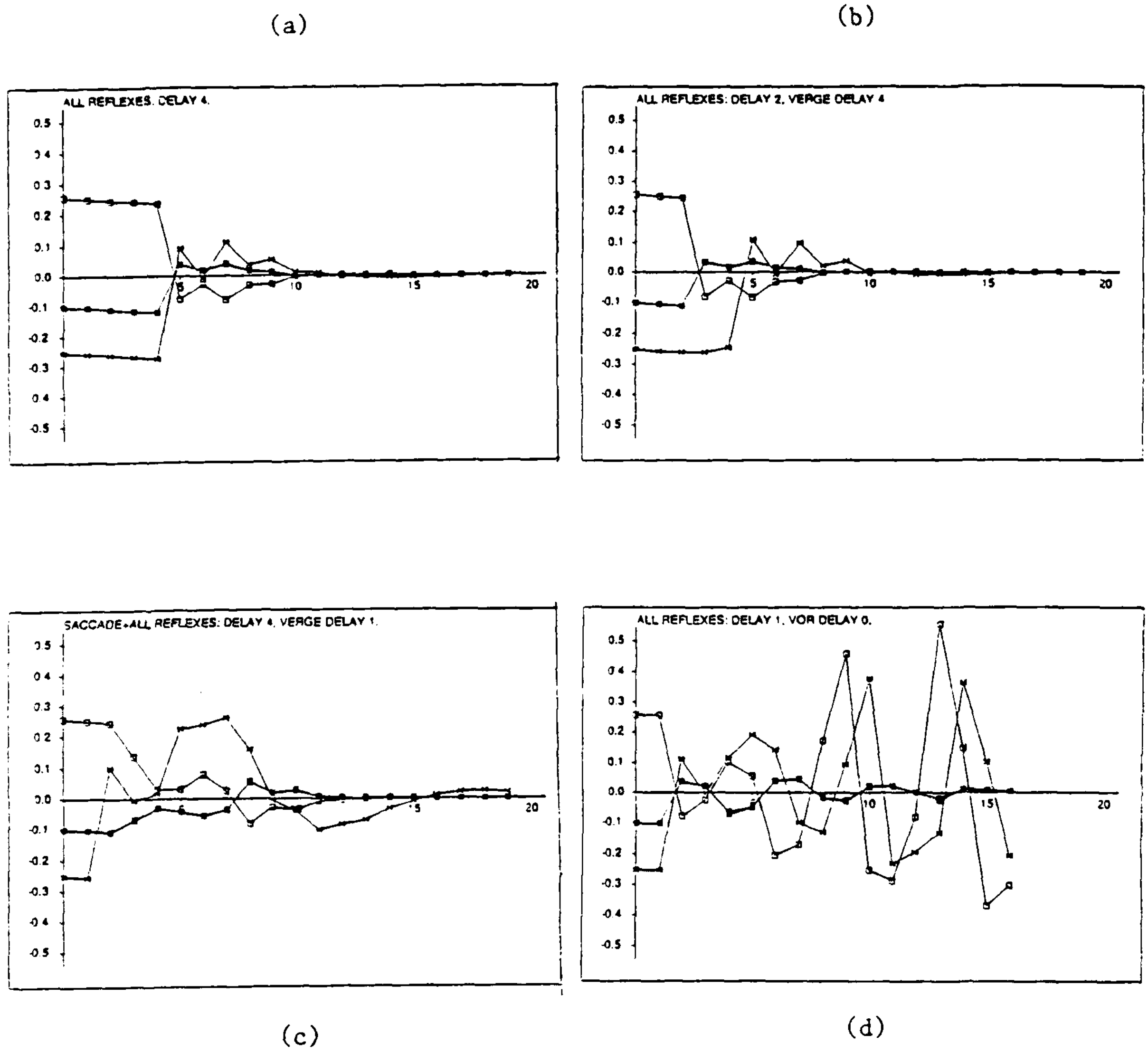


Figure 6: (a) To be compared with Fig. 2(d) and Fig. 5(a). The Smith predictor with independent control is stable with uniform controller delays. (b) Independent control also is stable with vergence control delay different. (c) Saccades induce transients but the system is still stable even if vergence delay different. (d) System is unstable if a non-vergence control, here VOR, has different delay from other non-vergence controls.

own delay T, and the delays of all the other controllers in the set {S} that share an output with it. Then each controller can use the following (interacting controls) algorithm. *Look ahead the maximum delay M of any controller in {S} and retrieve the predicted robot and control states for that time. Apply the control appropriate for these future states at (possibly future) time M-T.* This algorithm successfully copes with a different delay for each control (Fig. 7a).

An easy implementation of this algorithm that loses some flexibility is simply to increase the delay of all controls that share an output to be the maximum delay of any of their number and apply the independent delay control algorithm. Then all controls in the set look ahead as far as their slowest member, and act at the current moment. The resultant slowing of fast controls is of course suboptimal when they do not have to act in concert with slow controls.

Figures 7 and 8 show some experiments with interacting delay control, and introduce stochastic disturbances in the inputs and delays. The system is robust against sensor noise, or varying uncertainty in target location. The preliminary conclusion is that the system destabilizes with unpredictable delays when the outputs are changing relatively fast, but (of course) is less susceptible to unpredictable delays if the control outputs are only changing slowly.

DISCUSSION AND FUTURE WORK

SIMULATION AND REALITY

The goals for the simulator were to provide a kinematic and imaging model fairly close to that of the Rochester robot. The model has no dynamics, but neither does the robot from the point of view of the applications programmer; the current robot and motor control software hides this level. The simulator does seem adequate to illustrate the characteristics of different styles of control and to demonstrate the qualitative behavior resulting from control interaction, delays, and various forms of uncertainty. As the sophistication of the control technology at Rochester increases, a useful simulator would have to incorporate increasingly sophisticated models.

Likewise the simulator's exterior world and image-processing model is simple, consisting of a single point whose image is instantaneously and reliably (if noisily) found. To some extent this is also realistic, since it reflects the capability of frame-rate feature detection [Bro88], but it ignores the existence of more sophisticated operations or those with longer time-constants.

Simulation is likely to remain a basic tool in a real-time robotics laboratory, but as the control and visual environment gets sophisticated the simulations become slow and costly. The advent of cheap real-time hardware makes it increasingly practical to replace simulations with real-world experiments, which are more likely to yield relevant results.

COMPARISON WITH PRIMATE GAZE CONTROL MODELS

Because of its experimental accessibility, the simplicity of the plant involved, and the diverse collateral knowledge about the visual system, the gaze control system is the best-studied biological sensorimotor control system. The animal model most relevant to our robotic work is the primate, because of the close relationship of visual attention with fixation that arises with foveal (i.e. narrow-angle, high-resolution) vision. Gaze control in the cat and rabbit (and frog) is significantly different.

Knowledge of the primate gaze-control system might help provide insight to robot designers, and if the right hardware were available robotic equipment might be used to implement computational models of gaze control, thus providing an experimental facility complementary to the usual psychophysical and neuroscientific ones. The work described here is not yet dedicated to modeling biological systems, but nonetheless comparisions

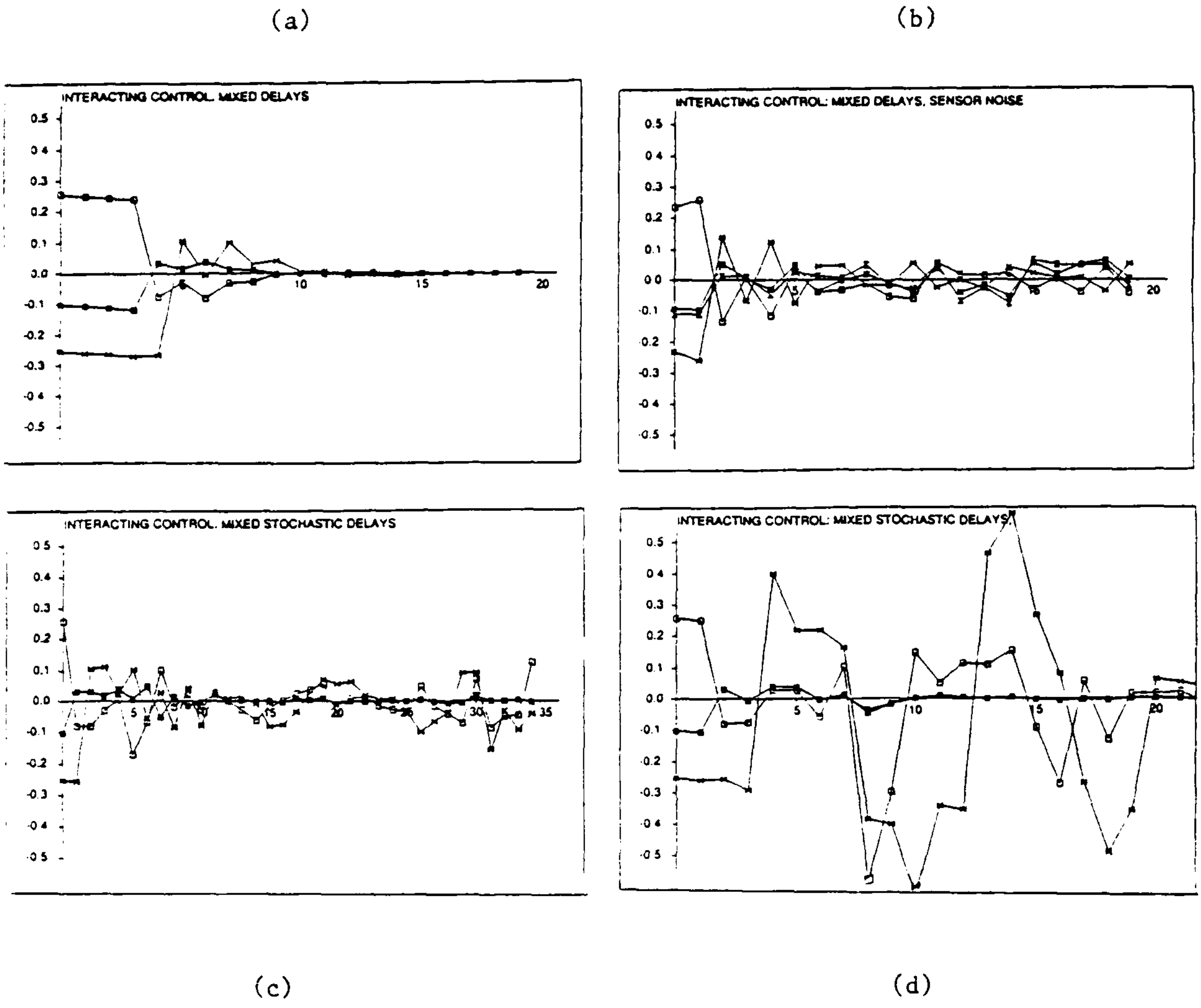


Figure 7: (a) The interacting control algorithm dealing successfully with a mixed set of delays. Here the longest non-vergence delay is three ticks, and the resultant behavior is that of a system whose non-vergence controls have a uniform delay of that amount. (b) Sensor noise (uniformly distributed disturbance of the target (x,y) location in each eye with $\sigma = 0.02$ in each dimension) does not affect stability, but causes excursions larger than its σ through the interaction of tracking and verging. (c) Here with probability .1 a control signal is delivered one tick early, and with probability .25 it is delivered one tick late. The system is on the verge of instability. (d) With same probabilities as in (c), more disturbances happen to occur early in the sequence when outputs are changing rapidly, destabilizing the system.

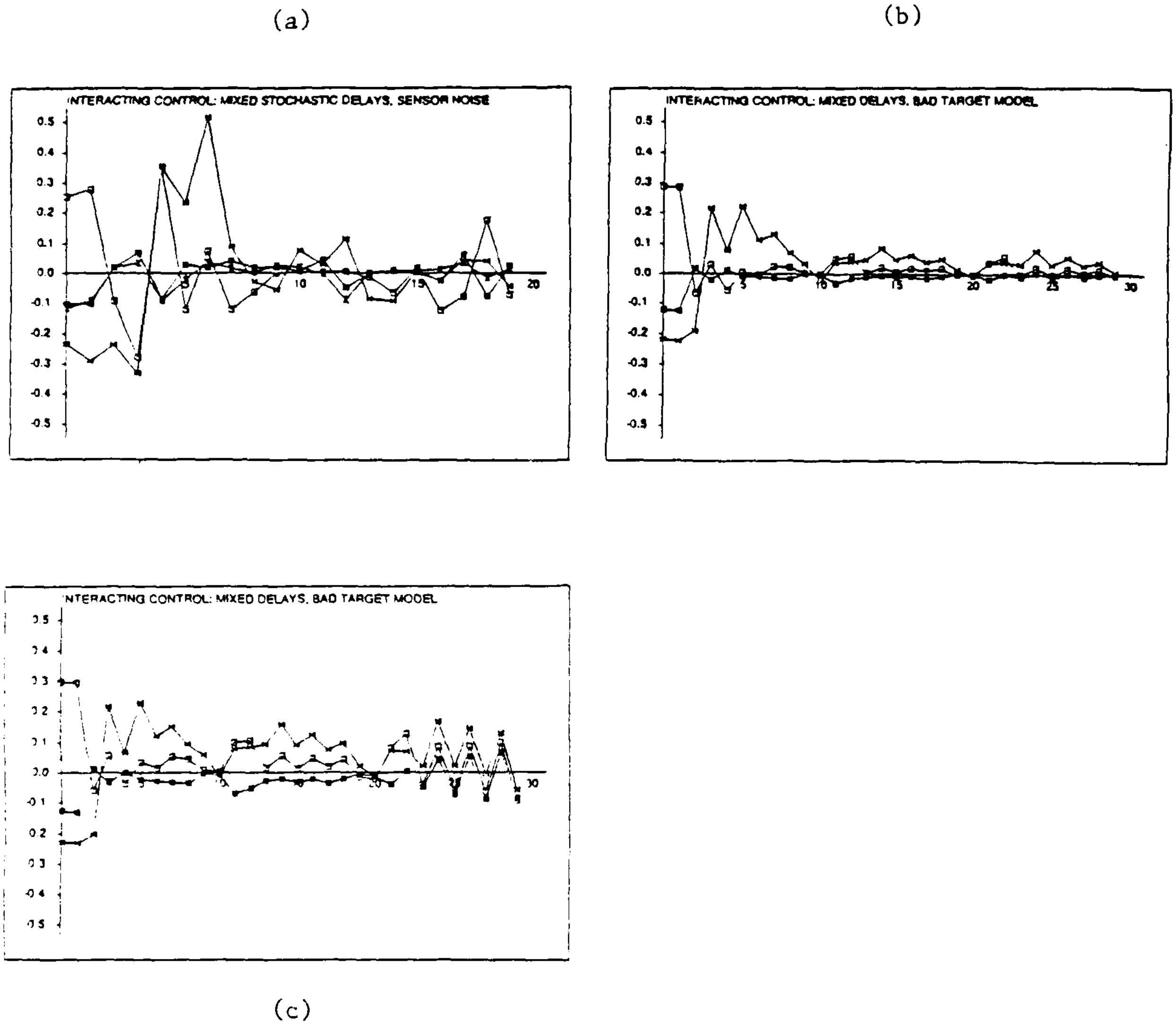


Figure 8: (a) Continuing from the previous figure, the previous sensor noise is added to the system along with the previous stochastic delays: the system is stable. (b) Here there is no noise (other than peripheral blurring), but the target model is wrong. The target is moving approximately perpendicular to the robot's motion instead of remaining static. The error periodicity of 10 ticks is interesting. (c) Here the situation is as in (b), but the target is moving faster, and toward the robot. As it gets close the controls cannot respond fast enough and the system destabilizes.

are inevitable, amusing, and possibly useful. This section is a very brief and admittedly selective sampling from the immense and rich (i.e. confusing and contradictory) literature on gaze and head control in biological systems. It seems fair to say that most of these systems interact, and that it is very difficult to lay down hard and fast rules about what individual systems can and cannot achieve.

Pursuit and Opto-Kinetic Reflex

The Opto-Kinetic Reflex (OKR) causes the eyes to follow a motion of the full visual field, and is driven (to first order) by "retinal slip", or optic flow. In primates the OKR comes in two stages, a faster (direct) and a slower (indirect), with the direct being more dominant in man. The smooth pursuit mechanism is to track small targets, and is often described as being driven by foveal retinal slip. Thus these two facilities are similar, and there is some thought that the direct part of the OKR response is just the smooth pursuit system [Col85].

The situation with smooth pursuit is anything but simple, however. It seems to be possible to pursue extra-foveal targets smoothly. Smooth eye movements cannot normally be induced without a smoothly-moving stimulus, but they persist after a target disappears, thus arguing that some form of prediction can excite the response [Eck83]. Smooth pursuit gain drops with stimulus velocity. Last, smooth pursuit in monkeys seems to be driven (in a large fraction of individuals) not just by velocity error but also by position and acceleration errors. Thus a model such as Young's (see below) that suggests a reconstructed target velocity is the control input (rather than a sensed optical flow) could be augmented with a broader range of error signals [LMT85].

The simulator has implemented both velocity control and position control with predictable results (compare Fig. 3(b) with Fig. 4(b)). Without position feedback, the system matches velocity and relies on saccades, which take place when position error goes over a threshold, for position control. There seems no advantage to this implementation unless optic flow velocity can be sensed directly, as opposed to position. For instance, if motion blur could be directly sensed, it would make a direct optic-flow velocity signal. Of course analysis of a particular motion-blur track could yield its centroid or endpoints, bringing us back to position control.

Vergence and Saccades

The primate vergence system is rather slow, and coupled to the focussing (accommodative) systems and the saccadic system. Vergence and accomodation are coupled pairwise, and the "near triad" is a reflex made up of these three systems, in which focus and vergence are both driven in the proper direction and faster than normal when a saccade from close to distant target (or the reverse) is made [Mil85].

Work with the Rochester robot has concentrated on "gross vergence", mediated through disparity computed between full-field images with variants of the cepstral filter [OP89]. The simulator described here is driven by horizontal disparity between the left and right target images. In the simulator, (which does not include focus) the cooperation of vergence and saccades is achieved simply, by the device of letting imaging, disparity calculation, and vergence reflex run during saccades. This method may or may not be nonbiological (as usual there is some dispute about the amount of visual processing that goes on during saccades). Its practical disadvantage is that it is inefficient: It is just as easy to have the saccade control both eyes. The only reason the current simulator does not run this way is that it is less interesting.

The saccadic system has a longer delay than smooth pursuit (120ms as opposed to 50 ms), reflecting its higher-level control origins. It can move the eye at 300 to 400 degrees/second. It is often modeled as a sampled-data system, kept stable by a latency and trigger mechanism that inhibits its firing again before the system has settled. In our robot system, saccades should not be needed for position control during tracking, and thus will be associated with shifts of attention, or at least of visual resource commitment.

In the experiments shown, the maximum saccade speed was limited but the maximum speeds for other reflexes were not (compare the .1 rad/tick saccade rate in Fig. 3(a) with the .3 rad/tick speed of the tracking and vergence in Fig. 2(d). Clearly the control should not be allowed to command unrealistic speeds, and the relative strengths of the outputs must be adjusted. In our simulation, the strictly "left eye dominant" implementation of saccades and of tracking is almost certainly an exaggeration of the ocular dominance effects in primates. Still, from a practical point of view it means that the necessary low-level vision computations do not need to be carried out in both eyes simultaneously.

The Vestibulo-Ocular Reflex

The Vestibulo-Ocular Reflex (VOR) stabilizes gaze bv counteracting commanded head movements with eye movements. It is the fastest visual reflex, with a delay of only approximately 16 milliseconds. It is an open-loop control, in the sense that vestibular sensor output is converted to eye muscle input and delivered through a path of approximately three synapses. It can be a high gain control (gain approximately 1): it can often exactly cancel out head motion effects. The VOR being open loop, there is a general problem of how it internally models the system it is controlling.

Research on the VOR has addressed the geometrical aspect of its modelling: the conversion of sensor signals in the coordinate systems of the semicircular canals to effector signals for the variously-placed eye muscles. Robinson [Rob85] models the geometrical transformations as 3x3 matrices operating on 3-vectors. Changing matrix components can accomplish adaptation, and the adaptation can be driven by stimuli such as retinal slip (indicating a failure of the reflex) without explicitly modelling the sensorimotor system. Pellionisz [Pel85,PP88] uses tensors to model the differing transformation properties of the sensory and motor vectors and transformations, and addresses the problem of underdetermined control of the many muscles that accomplish eye and head movements by the relatively small number of sensor dimensions.

The VOR's input originates in the linear and angular accelerometers of the otolith organs and semicircular canals. They have very short time constants, but the VOR operates correctly for slow velocities. This leads to the postulation of a "velocity storage mechanism" that integrates the output of the accelerometers and makes the resulting velocity signal available for control (e.g. [RC85]).

Other VOR work addresses its time-dependent behavior: its gain and phase-lag characteristics under different conditions (e.g. several papers in [BJ85]). Much of the VOR's behavior can be explained as parameter variation among its gain, bias, and time constants. Miles *et al.* [MOL85] develop a multi-channel model to explain VOR's ability to cope with the frequency-dependent output characteristics of the sensors, with frequency-selective adaptation properties of the VOR itself, and with other adaptive properties of the VOR. This work presents explicit transfer functions for the semicircular canals, the oculomotor plant, the velocity storage mechanism, and the neural channels that convert head velocity estimates to motor outputs. The channel model is linear and can be stated as a lumped-parameter linear system, but the channels make it easier to identify which gains must be changed to reduce system errors.

A basic aspect of the VOR is its adaptability. The reflex adapts over time to changes in the optical system (e.g. artificially induced dysmetria) [Rob85]. The VOR interacts with other reflexes and the stimuli that evoke them. For example, large-field rotations that elicit the OKR have an interesting effect. If they are slow, they bias the VOR (and the opto-kinetic system) in the same direction, which tends to cancel the movement effect. If they are fast, they induce effects in the opposite direction, which may be interpreted as ignoring the movement effect [Col85]. VOR gain can be depressed from 1.0 to 0.1 by training that involves no visual input (subject imagines tracking a target attached to head while moving head in the dark), and is likewise significantly affected by verbal instructions and other seemingly unrelated activities (such as mental arithmetic) [JB85].

Adaptation and modeling can come together in VOR behavior that adapts to repetitive patterns (a perhaps familiar example is disembarking from a longish sailing journey). One way to achieve this capability is through a "pattern storage" mechanism that effectively produces and uses a model of the outside world. Some workers are attracted to this idea, others seem to think it is unnecessary and are explicable by, for instance, channel adaptation.

What has all this to do with a robotic VOR? Many of the issues mentioned above can be made to vanish.

We may know the relation of the sensor output to the desired motor output if we decide to model the robot and head kinematics accurately. (In fact in the simulation, the robotic VOR makes several approximations, including a "spherical" geometry for the camera rotation axes, a small-angle approximation, and others.) We can sense velocities directly or even actively monitor the relevant control signals we need to cancel. The fundamental issues that still need significant work involve adaptation and interaction. Adequate understanding of these issues would not only give the robot system the efficiency exhibited by natural systems, but could mean that such exercises as accurate kinematic modeling would become unnecessary.

Head Control

There is less written on head control than on gaze control, but a good recent collection of work exists [PR88]. There are various head stabilization reflexes, some tied to optical stimulation. The relation of head control strategies to the evolution of particular brain mechanisms and the existence of foveate vision is explored by Roucoux and Crommelinck [RC88]. Some fairly detailed biomechanical head models exist, and head movements have been investigated from the point of view of optimal control theory. Head movements can be quite rapid (600-700 degrees/second) and are part of normal long-distance saccades in primates. Thus the saccadic and head-control system work together to achieve gaze redirection. There has been some work here (e.g. [Gui88]) indicating that head movements can take place at differing times relative to saccades. Typically, they lead or lag depending on whether the target location is predictable or not.

This coupling of head and eye movements is clearly more sophisticated than the compensatory reflex implemented in the simulation, which is not coupled to saccades at all and which must lag eye movements since it is only driven by eye positions. Thus more work needs to be done if we are to achieve the increased rapidity of gaze redirection that arises when both head and eyes are moved in a coordinated way.

Another Model of Delay Control

The control scheme implemented in this simulation, the Smith predictor, differs from a scheme seemingly first proposed in a gaze-control context by Young, taken a step further by Robinson, and used recently in robotic gaze-control for an agile, two-eyed robotic head at Harvard University [CF88].

Young [You77] wanted to explain how smooth pursuit avoided instability in the presence of two difficulties that apply if tracking is modeled as a pure negative feedback system. First, the error, and thus control, signal is zero when accurate tracking is achieved; this should send eye velocity transiently to zero. Second, tracking performance is better than it should be given the delays in the control loop and the time constants of the processes. His proposal is that the system tracks not the retinal image, but a neural signal that corresponds to target motion (in the world).

In 1971 (for a recent reference, applied to saccadic, tracking, and limb control, see [Rob88]) Robinson proposed a mechanism to implement Young's idea. In the negative feedback system the eye velocity is fed back and subtracted from the target velocity (with some delay). If the eye is in the process of tracking, then the target velocity is the sum of the eye velocity (with respect to the head) and the target's retinal velocity (its velocity with respect to the eye). But the latter is just the error signal resulting from negative feedback. Thus an estimated target velocity signal can be constructed by positively feeding back the commanded eye motion into the control loop, delayed to arrive at the proper time to combine with the error term produced by negative feedback. This mechanism not only provides a signal based on the target's true motion, but it cancels the negative feedback and thus removes the possibility of oscillations.

Robinson's scheme is related to the Smith controller shown in Figure 1 in the following way. In Figure 1, the signal at E is an error signal, and the one at D is a difference of error signals that is zero when perfect tracking is taking place. This difference of errors is a delayed (but consistent) error signal that is added to the predicted error signal in the non-delayed path C. The controller in Figure 1 tries to drive errors to zero. To change Figure 1 to Robinson's scheme, delete path C and remove the modelled world and sensor from the lower half of the block diagram. Then path B carries the simulated plant, not the simulated error. Path E still contains error, but path D now contains a prediction, or reconstruction, of the world state. Thus the controller now must treat the signal at D as a set point to be achieved through open-loop methods, not as an error. Robinson proposes parametric adaptive control (in the form of two related gains) to provide adaptative capability should the open loop yield the wrong results.

There are thus some similarities between the two schemes, but the underlying control philosophies are rather different. In paricular, losing the power of negative feedback is a large sacrifice that the roboticist may not need to make. The Smith predictor control system keeps the advantage of feedback control (running on the modelled world and plant). There are many methods of estimation, observation, and prediction of world, sensor, and plant used in modern control theory, and thus the Smith model allows for flexibility in the assumptions underlying its predictions.

FUTURE WORK

We plan to supply more quantitative model parameters, and to try to model the spatial and temporal scales that actually apply in the laboratory. Sensitivity analysis will be undertaken to quantify the effects of various disturbances, especially the problem of unpredictable delays.

We plan to integrate some of the existing Kalman filtering tracking utilities [Bro89,BF88] to perform estimation of the target's state. Also we may explore estimation techniques [Gel73,Ber76,Eyk74] instead of simulation techniques to predict the state of the plant.

The simulated system can support other relevant aspects to the control problem, including the important one of adapting to changes in the plant. In other work, we have implemented "the MIT rule", which is a gradient descent method similar to back-propagation learning in neural nets, to learn part of the robot head geometry. In a way this learning system acts like another control system, with inputs the discrepencies between expected and observed target motions given eye motions, and outputs are parameters to the modeled plant (in this case, lengths of links in the head kinematic chain).

Implementation of an increasingly sophisticated gaze control system on the Rochester robot should take place over the next few years. We anticipate substituting a Butterfly Parallel Processor with multiple input and output ports for the central controller of the system.

Acknowledgements

This work is funded by the DARPA U.S. Army Engineering Topographic Laboraties Grant DACA76-85-C-0001 and the Air Force Systems Command (RADC, Griffiss AFB, NY) and Air Force OSR Contract F30602-85-C-0008, which supports the Northeast Artificial Intelligence Consortium. The assistance of Prof. J. Michael Brady, Dr. Hugh Durrant-Whyte, Dr. Ron Daniels, and the RRG research and administrative staff is gratefully acknowledged.

References

[Ber76] Dimitri I. Bertsekas. *Dynamic Programming and Stochastic Control.* Academic Press, 1976.

[BF88] Y. Bar-Shalom and T.E. Fortmann. *Tracking and Data Association.* Academic Press, 1988.

[BJ85] A. Berthoz and G. Melvill Jones. *Adaptive Mechanisms in Gaze Control: Facts and Theories.* Elsevier, 1985.

[BO88] D. H. Ballard and A. Ozcandarli. Real-time kinetic depth. In *Second Int. Conf. on Computer Vision*, November 1988.

[BR88] C. M. Brown and R. Rimey. *Kinematics, Coordinate Systems and Conversions for The Rochester Robot.* Technical Report 255, University of Rochester, September 1988.

[Bro88] C. M. Brown. *The Rochester Robot.* Technical Report 257, University of Rochester, September 1988.

[Bro89] C. M. Brown. Kalman filtering for tracking and control. In *DARPA Image Understanding Workshop*, June 1989.

[CF88] J. J. Clark and N. J. Ferrier. Modal control of an attentive vision system. In *Second Int. Joint Conference on Computer Vision*, November 1988.

[Col85] H. Collewijn. Integration of adaptive changes of the optokinetic reflex, pursuit and the vestibulo-ocular reflex. In A. Berthoz and G. Melvill Jones, editors, *Adaptive Mechanisms in Gaze Control*, Elsevier, 1985.

[Eck83] R. Eckmiller. Neural control of foveal pursuit versus saccadic eye movements in primates – single-unit data and models. *IEEE Trans. on Syst., Man, and Cyber.*, SMC-13(5):980 – 989, Sept./Oct. 1983.

[Eyk74] Pieter Eykhoff. *System Identification: Parameter and State Estimation.* Wiley and Sons, 1974.

[Gel73] Arthur C. Gelb. *Applied Optimal Estimation.* The MIT Press, 1973.

[Gui88] D. Guitton. Eye-head coordination in gaze control. In B. W. Peterson and F. J. Richmond, editors, *Control of Head Movement*, Oxford University Press, 1988.

[JB85] G. Melvill Jones and A. Berthoz. Mental control of the adaptive process. In A. Berthoz and G. Melvill Jones, editors, *Adaptive Mechanisms in Gaze Control*, Elsevier, 1985.

[LMT85] S. G. Lisberger, E. J. Morris, and L. Tychsen. Visual motion processing and sensory-motor integration for smooth pursuit eye movements. In A. Berthoz and G. Melvill Jones, editors, *Adaptive Mechanisms in Gaze Control*, Elsevier, 1985.

[Mar79] J. E. Marshall. *Control of Time-Delay Systems.* Peter Peregrinus Ltd., 1979.

[Mil85] F. A. Miles. Adaptive regulation in the vergence and accommodation control systems. In A. Berthoz and G. Melvill Jones, editors, *Adaptive Mechanisms in Gaze Control*, Elsevier, 1985.

[MOL85] F. A. Miles, L.M. Optican, and S. G. Lisberger. An adaptive equalizer model of the primate vestibulo-ocular reflex. In A. Berthoz and G. Melvill Jones, editors, *Adaptive Mechanisms in Gaze Control*, Elsevier, 1985.

[OP89] T. Olson and R. Potter. Real-time vergence control. In *Computer Vision and Pattern Recognition 1989*, June 1989.

[Pel85] A. J. Pellionisz. Tensorial aspects of the multidimensional approach to the vestibulo-oculomotor reflex and gaze. In A. Berthoz and G. Melvill Jones, editors, *Adaptive Mechanisms in Gaze Control*, Elsevier, 1985.

[PP88] A. J. Pellionisz and B. W. Peterson. A tensorial model of neck motor activation. In B. W. Peterson and F. J. Richmond, editors, *Control of Head Movement*, Oxford University Press, 1988.

[PR88] B. W. Peterson and F. J. Richmond. *Control of Head Movement.* Oxford University Press, 1988.

[RC85] T. Raphan and B. Cohen. Velocity storage and the ocular response to multidimensional vestibular stimuli. In A. Berthoz and G. Melvill Jones, editors, *Adaptive Mechanisms in Gaze Control*, Elsevier, 1985.

[RC88] A. Roucoux and M. Crommelinck. Control of head movement during visual orientation. In B. W. Peterson and F. J. Richmond, editors, *Control of Head Movement*, Oxford University Press, 1988.

[Rob85] D. A. Robinson. The coordinates of neurons in the vestibulo-ocular reflex. In A. Berthoz and G. Melvill Jones, editors, *Adaptive Mechanisms in Gaze Control*, Elsevier, 1985.

[Rob88] D. A. Robinson. Why visuomotor systems don't like negative feedback and how they avoid it. In M. A. Arbib and A. R. Hanson, editors, *Vision, Brain, and Cooperative Computation*, MIT Press, 1988.

[Smi57] O. J. M. Smith. Closer control of loops with dead time. *Chemical Engg. Prog. Trans.*, 53(5):217-219, 1957.

[Smi58] O. J. M. Smith. *Feedback Control Systems*. McGraw-Hill, 1958.

[You77] L.R. Young. Pursuit eye movement - what is being pursued? *Dev. Neurosci.: Control of Gaze by Brain Stem Neurons*, 1:29–36, 1977.

Towards Autonomous Mobile Robot Navigation[1]

Claude Fennema, Allen Hanson, Edward Riseman
Department of Computer and Information Science
University of Massachusetts
Amherst, MA 01002

ABSTRACT

The UMass Mobile Robot project is investigating the problem of intelligent navigation of an autonomous robot vehicle. Model-based processing of the visual sensory data is the primary mechanism used for obstacle avoidance, movement through the environment, and measuring progress towards a given goal. This paper describes our current approach to goal-oriented navigation through a partially modeled, unchanging environment which contains no unmodelled obstacles.

The navigation system integrates perception, planning, and execution of actions. Of particular importance is that the planning processes are able to reason about landmarks that should be perceived at various stages of plan execution. Correspondence between image features and expected landmark locations are used at several abstraction levels to ensure proper plan execution. Experiments in this and three companion papers demonstrate the performance of the various components within the navigation system.

I. INTRODUCTION

The UMass Mobile Robot project is investigating the problem of enabling a mobile automaton to navigate intelligently through indoor and outdoor environments. At the foundation of our work is the premise that higher-level vision beyond the first stages of sensory processing will greatly benefit from, and in many cases require, the use of knowledge and models of objects in the environment. Thus, model-based processing of the visual sensory data is the primary mechanism used for obstacle avoidance, movement through the environment, and measuring progress towards achieving a given goal.

Our mobile robot, called Harvey, is a Denning platform ultimately intended to navigate through offices, hallways, and university grounds as it carries out commands such as "Fetch the book" or "Bring this to Allen". Since this is a rather formidable task, we have developed a research plan that will be carried out in stages of increasing generality and functionality. In the early phases of this research, we wish to balance generality with setting sufficient constraints on the initial research goals to be achievable. Our initial experiments focus on robust goal-oriented navigation through a partially-modeled, unchanging environment that does not contain any unmodelled obstacles.

If robust autonomous navigation can be achieved in this restricted domain, then a variety of challenging problems can be considered as the constraints are eased on the assumed knowledge about the environment. These problems include: navigation in a partially known environment with obstacles, navigation in the presence of independently moving objects, and exploration of an unknown environment to learn a model in order to support future model-directed navigation. This paper, however, describes the current UMass approach to the initial problem domain of robust navigation in a partially-modelled

[1] This work was supported in part by the Defense Advanced Research Projects Agency under contract numbers F30602-87-C-0140, DACA76-85-C-0008, and DACA76-86-C-0015, and by the National Science Foundation under grant number DCR-8500332.

environment, and our experiments in testing an implementation of such a system.

I.1 Related Mobile Robot Research

We begin with a brief survey of previous mobile robot research; other relevant research will be addressed in the sections discussing particular system modules. The Carnegie-Mellon NAVLAB (Kanade, Thorpe et al. 1986; Shafer, Stentz et al. 1986) and the Martin-Marietta ALV (Lowrie, Thomas et al. 1985) are systems that can move down a path or road or navigate off-road terrain, but the processing has been restricted to simple goals, such as controlling the vehicle relative to the sides of the road, or avoidance of major obstacles such as trees. Recent demonstrations of these systems have been quite interesting, but a laser range sensor providing depth information played a significant role in the obstacle avoidance capabilities.

Brooks (Brooks 1986) has an unusual demonstration of low-level behaviors and motor activity to allow a relatively inexpensive robot to wander in an unknown environment carrying out some purposeful activity, but this work has not yet focused on the achievement higher-level goal-oriented navigation tasks, and does not make use of models of the environment.

Dickmanns and Graefe (Dickmans and Grafe 1988a; Dickmans and Grafe 1988b) have developed techniques for using image features in a real time feedback control loop to control the motion of a car on the autobahn. In the system we develop in this paper their techniques could serve as part of the function we term "action level servoing". The approach described here, like Dickmans and Graefe, accomplishes servoing by tracking image features, but here the tracking features are constructed from landmarks which have been selected from a knowledge base.

Due to the complexity of visual perception, autonomous navigation projects, such as those cited, have utilized only limited visual processing, either in terms of the features extracted from the environment, or the modeled set of objects to be recognized in the environment, or both. This is not meant to be a serious criticism, but rather serves as an observation for the reader who does not recognize the extreme complexity of the problems of vision and autonomous navigation in natural outdoor domains.

Recently, Faugeras (Toscani and Faugeras 1987) used more sophisticated vision algorithms involving stereo to derive depth in an office scene. Depth information was extracted from a stereo pair, the robot was moved some distance, and a second stereo pair was used to derive depth and the associated motion. Again, this effort does not represent a full robot navigation system, and made no use of high-level models.

I.2 Overview of System Modules

The processing modules that provide the basic functional capabilities for our mobile robot system are briefly outlined below. There are many possible control strategies and system organizations that can be imposed on top of these modules to support effective mobile robot navigation. In Section III, we briefly outline one such control strategy.

Modelling the 3D Environment (Connolly 1989; Connolly and Weiss 1989) - Geometer is a solid modelling package that was jointly developed at the University of Massachusetts and General Electric Corp. The CAD system provides tools for representing knowledge of shape in an annotatable hierarchy.

Planning (Fennema, Hanson et al. 1989; Fennema, Riseman et al. 1988)- Tasks (or goals) are translated by a command interpreter and decomposed by a hierarchical problem solver into a sequence of milestones and proposed actions. Plans are developed depth-first, with less detail away from the current task; task failure triggers dynamic replanning.

Monitor Plan Execution (Fennema, Hanson et al. 1989; Fennema, Riseman et al. 1988) - Plans are executed in a repetition of two operations: recognize milestone and execute primitive action. Each milestone is constructed from a perceivable 3D landmark derived from the model. Finding the projection of the landmark in the image signifies a successful completion of the associated action.

2D Line Model Matcher (Beveridge, Weiss et al. 1989a; Beveridge, Weiss et al. 1989b) - This module finds a best match and fit of a given 2D line model to a subset of data line segments that may have been fragmented, skewed, omitted, etc. during low-level processing. A search through the plausible symbolic correspondences between model and data lines is performed, and the optimum 2D translational and rotational fit for each is computed as a closed-form solution.

3D Pose Refinement (Kumar 1989) - Given correspondences between a set of points and lines in a 3D model and a 2D image, the 3D camera location and orientation is computed as an optimization procedure. In addition, uncertainty in the output parameters as a function of the variance of the noise in the input parameters is provided.

In addition to these modules, several basic vision modules have been developed. These modules include a fast line finder (Kahn, Kitchen et al. 1987) derived from a straight line algorithm developed by Burns (Burns, Hanson et al. 1986), a histogram based region segmentation algorithm (Beveridge, Griffith et al. 1989), an algorithm for determining subpixel line placement given an image line, and a local template correlation mechanism (Fennema, Hanson et al. 1989).

II. GEOMETER AND MODELS OF THE ENVIRONMENT

II.1 Geometer

Models of the vehicle's environment are built using Geometer, a three-dimensional solid modeling package developed jointly by UMass and the GE Research and Development Center (Connolly 1989; Connolly and Weiss 1989). Geometer is implemented in LISP and is oriented towards image understanding (although it has many other potential applications). It currently runs on several types of workstations, including the Symbolics LISP machines, TI Explorers, Vax workstations, and Sun workstations. Refer to (Connolly and Weiss 1989) in this proceedings for additional information about Geometer.

Objects in Geometer are represented in an annotatable hierarchy:

World → Object → Faces → Edges → Vertices.

In Geometer, the language of simplicial complexes in algebraic topology (Eilenberg and Steenrod 1952; Greenberg and Harper 1981) has been adapted for describing surfaces. It provides generality and an explicit representation of edges, vertices, and faces. Each of these serve as a type of geometric primitive, and can be parameterized as a smooth function from a point, unit interval, and triangle to $\mathbf{R}^3$ respectively. Surfaces are constructed as the union of these primitives, and are denoted by a sum of simplices. This representation produces a triangulation of the surface, where the triangles are not necessarily planar.

II.2 Constructing Environmental Models

The system begins with an accurate, but incomplete, model of the world implemented in Geometer, augmented by the locale structure described in the next section. We have constructed a 3D model of portions of the interior of the UMass Graduate Research Center, as well as a portion of the campus surrounding the building. The outdoor model (shown in Figures 1 and 2) includes buildings (windows, doors, pillars, etc.), sidewalks, lamp posts, telephone poles, and most of the significant objects in the area. This model has been annotated with properties of objects and surfaces which are useful to the planning and vision routines used by Harvey.

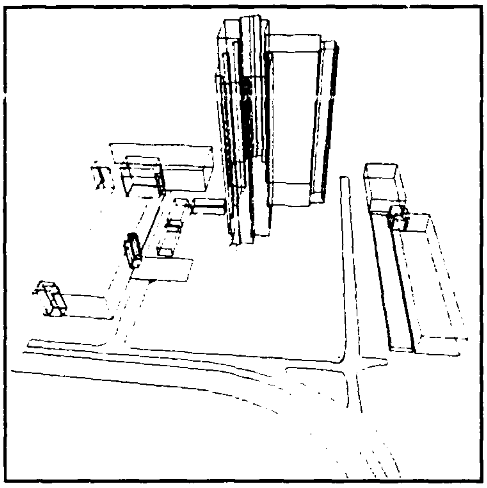


Figure 1. Geometer model of the area around the Graduate Research Center used in the experiments.

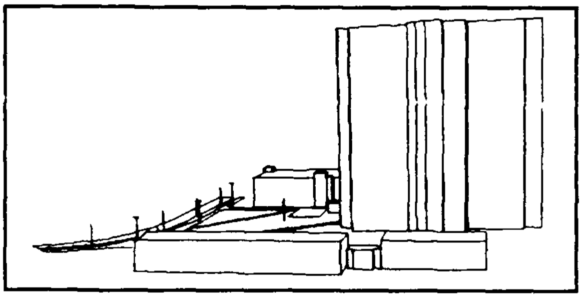


Figure 2. A more detailed Geometer model of the same areas shown in Figure 1 with hidden lines removed. Note that additional landmarks, such as telephone poles, have been added.

The construction of an accurate 3D model of an environment is a fairly difficult job. The first attempt involved digitizing data from engineering blueprints using a bit pad (digitizing tablet). This method is quite error prone given the spatial resolution of the bit pad, since the blueprints were drawn to a scale of 40 feet to an inch We found errors of up to 10 feet in the 3D model constructed in this manner. In the second attempt, theodolites were used to survey the landmarks. This method, while accurate, is very time consuming. As a check, some of the theodolite data was verified by direct measurement. On the average, the measured distances matched with the surveyed distances within 0.2 feet.

II.3 Locales

The model of space in this system plays a rather central role in most of the robot's activities. During planning, for example, the model is used to construct routes. Consequently, the concept of doorways, portals, exits, and entrances must be represented. During plan monitoring, the model is used in a top-down fashion to control visual perception by specifying what is to be "seen" and where to "look for" it. In this situation, only the space within the perceptual field of view of the robot is relevant. If the robot gets lost, the world model is used as a means for localizing it within the environment. Space should be represented and organized in a way which simplifies these tasks.

Conceptually, our view of the organization of space is inspired by the topological notion of a neighborhood. Hierarchically organized neighborhoods serve to successively localize a point to a finer resolution. We use this concept as a means for localizing the agent (robot) by associating with each neighborhood a means for determining whether or not the agent is inside it. This neighborhood-test pair is called a 'locale'. Locales impose an organization on 3D space and partition it into convenient subspaces that are used for planning and robot localization.

A locale is represented by a data structure that captures its neighborhood-like properties via a 3-D shape description of the locale and a contained-by hierarchy as shown in Figure 3. Each locale also contains additonal information, such as its shape descriptors as shown in Figure 4. From this locale data structure, it is possible to construct a test to determine whether or not the agent is in a particular locale and to pick landmarks to act as milestones.

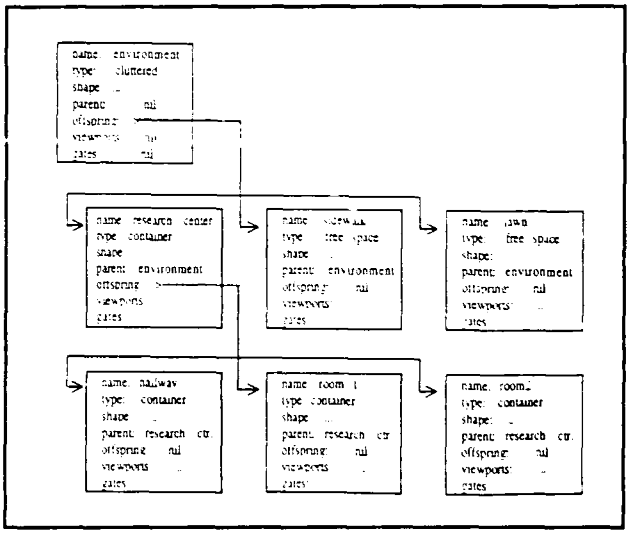


Figure 3. Locales are subspaces of the environment which are organized into a hierarchy by set inclusion. This simplified example shows three levels of locales representing the Graduate Research Center environment.

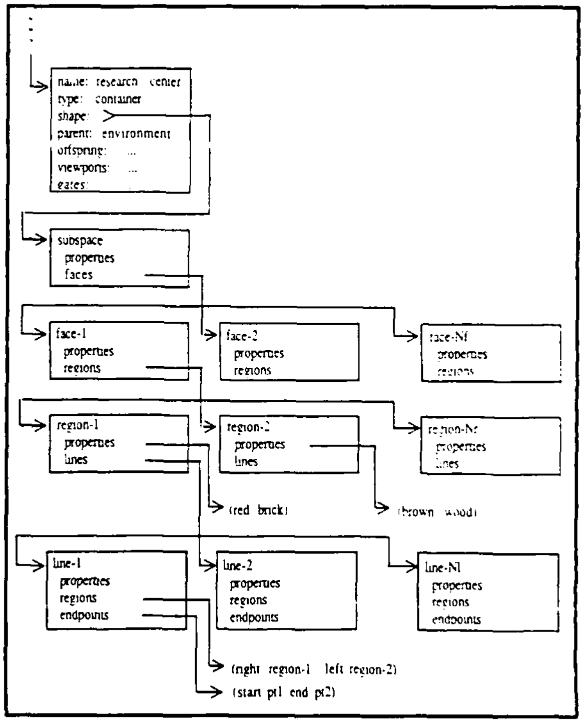


Figure 4. The actual shape descriptions of each locales is a hierarchy of geometric entities defined in Geometer. Shown are the entity properties used during perceptual reasoning to construct landmarks

III. A BRIEF LOOK AT PLANNING AND CONTROL

Each task given to Harvey is translated by a command interpreter and problem solver which ultimately produces a set of navigational goals. The execution of these goals is accomplished by a tight interweaving of planning, perception, and action, orchestrated by a dynamic planning and execution scheme (Fennema, Hanson et al. 1989; Fennema, Riseman et al. 1988) called "plan-and-monitor". This subsystem works with plans, each represented as a sequence (M0 A1 M1 ... AG MG) of milestones (Mk) and proposed actions (Ak). Milestones are used to verify the successful completion of a particular phase of the plan. They are composed of 3D landmarks (perceivable physical events) and their expected location with respect to the robot at the completion of the appropriate phase of the plan.

As a plan is executed milestones must be verified (usually visually) before the next action of the plan can be executed. For example, if the sequence of milestones up to M7 have been perceptually verified to be in the proper position in the image (i.e. within the acceptable error bounds), this means that actions A1, ..., A7 have been successfully completed, and it is appropriate to take action A8. If M7 cannot be verified, then the plan must be modified. In this way milestones allow the progress of the plan to be monitored, and trigger replanning before the next action is taken when perception and milestone do not agree(Fennema, Riseman et al. 1989). Complex actions and tasks also trigger replanning in order to refine them into a plan subsequence of milestones and primitive actions which can be directly executed by the hardwar.

The plan-and-monitor executive directs planning, perception, and execution in such a way as to dynamically modify and refine the plan to fit the actual results of each action and the details of the perceived environment. The principal activities involved in this process are: planning, milestone recognition, determination of location, and execution of primitive actions. This interweaving of perception, planning and action makes specific what task is expected of perception, and provides a means for focusing the knowledge available for that purpose. The results is a distribution of perception and perceptual reasoning into all aspects of navigation. Route planning uses perceptual reasoning to select appropriate perceptual milestones; plan progress is measured using perception; perception is used to relocate the robot when a milestone is not recognized; and during the execution of primitive actions, low-level perceptual feedback is used to keep the robot on the expected trajectory. The different levels of control all use model-directed vision and compare what is sensed to what is expected, issuing corrective commands to minimize any difference.

Plan execution depends upon the recognition of milestones. The difficulty of using vision to perform this task in a reliable and general manner has encouraged us to attack this problem in two ways. Both methods use model-directed processing by comparing restricted perceptual processing to what is expected if the robot's motor actions are correct. The next section describes a type of low-level perceptual servoing used during execution of primitive actions. Section 5 describes a more complex method for matching models to landmarks and refining the position of the robot based on these matches.

IV. EXECUTING PRIMITIVE ACTIONS: PERCEPTUAL SERVOING

Navigation goals are ultimately translated into primitive actions which can be directly executed by the robot vehicle; in the case of the Denning platform, these are (MOVE distance) and (TURN angle). Even at this primitive action level, however, execution errors are probable. As the robot rolls along an environmental surface a slippery spot, a bump on the surface, or even a bulge in its tire may throw it off course, causing inaccurate execution.

It is possible to reduce the error incurred when executing a primitive action by *servoing* on prominent visual features in the environment. Using information obtained from the measured discrepancy between where the features should be and where they actually are, it is possible to determine the corrective action required to bring the positions into agreement. This *action level* or *perceptual servoing* has the effect of locking the robot onto a trajectory which improves the accuracy of the primitive actions over that which would be obtained without servoing.

In order to determine the usefulness of servoing, a simple version was implemented that used correlation to measure the deviation of actual motion from intended motion. Several experiments, both with and without correlation servoing, were run. In the experiments Harvey was to roll along a straight line 40 feet long, marked on the floor of a Graduate Research Center hallway For the experiments in which servoing was used, an artificial target was placed on a door at the end of the hallway, since the Geometer model of the interior of the building was not complete. The target was a circle approximately eight inches in diameter with two opposing black quadrants and two opposing white quadrants. The robot's goal was to move down the corridor directly towards the target. To determine course deviation, the vehicle was stopped every two feet and its deviation from the marked line was measured.

The experiment was run a number of times; the results in Table 1 represent the best one in the sense that the unservoed results represent the smallest deviations encountered during the trials. The z-axis referred to in the table is the line the vehicle is following with z=0 defined as the starting location. The x-axis is a line perpendicular to the z-axis and pointing to the right. Total distance traveled is z-unservoed and deviation is x-unservoed. Even after a rather painstaking set up procedure the vehicle wandered over two inches from the line during a 20 foot motion. Other runs resulted in as much as a foot deviation in unservoed mode. Most of the trials in unservoed mode were stopped at around 20 feet because the vehicle was significantly off course and the total deviation was increasing. In contrast, in servoing mode the vehicle stayed within .3 inch of the line for 38 feet. It is worth noting that in both experiments the actual distance covered was considerably less than the intended distance. It is consistently short by a constant factor (to 3 decimal places), due to inaccurate calibration of the hardware.

Table 1. Results from one experiment
(All measurements are in inches)

intended-z	unservoed-z	servoed-z	intended-x	unservoed-x	servoed-x
24.	22.6	22.8	0.0	0.0	+0.13
48.	45.5	45.7	0.0	-0.3	+0.13
72.	68.3	68.5	0.0	-0.4	+0.13
96.	90.9	91.3	0.0	-0.6	+0.13
120.	114.2	114.3	0.0	-0.7	+0.06
144.	136.3	136.9	0.0	-1.1	0.00
168.	158.2	159.5	0.0	-1.3	-0.13
192.	181.8	182.2	0.0	-1.8	-0.13
216.	204.6	205.0	0.0	-2.0	-0.38
240.	228.3	227.7	0.0	-2.1	-0.25
---.	---.-	---.-	-.-	-.-	-.--
480.	---.-	456.0	0.0	-.-	0.0

The results of these experiments are encouraging and support the idea of action level perceptual servoing over reasonably short navigation legs; additional results for (MOVE distance) as well as servoing results for (TURN angle) are presented in (Fennema, Hanson et al. 1989). Once the Geometer model of the building interior is complete, similar experiments will be performed using actual geometric features rather than the artificial target. When weather permits, the vehicle will be moved outdoors and the Geometer model described in Section II will be used to determine the effect of terrain cover and topography on servoing accuracy.

V. RECOGNIZING AND USING 3D LANDMARKS

Recognition of 3D landmarks involves matching an object model to data extracted from an image, and this task has two parts: a)determining the correct correspondence between object features and image features and, b)determining the position of the object with respect to the camera. We refer to the former task as 2D model matching (Section V.1) and to the latter as 3D pose refinement (Section V.2) These sub-tasks are interdependent, since an object's position relative to the camera in 3D space cannot be determined without determining a correspondence to image features, while the correct correspondence depends on the object's 2D appearance and hence its relative position and orientation in space.

V.1 2D Model Matching

In contrast to the approach developed by Lowe for the SCERPO system (Lowe 1985; Lowe 1987) we have chosen to separate the 2D processing of model-to-image matching from the 3D optimization process for computing the camera pose once the correspondences between model and image are completed. Thus, we restrict ourselves in this section to the problem of matching a 2D model to a set of fragmented, skewed, and missing line segments, a rather challenging perceptual organization problem. The model line to image line correspondences determined from this 2D matching method are used as the input to the 3D pose computation discussed in the next section.

We believe that there are strong incentives to solve as much of the identification problem as possible via processing in the 2D image space. The combinatorics of establishing correspondences between object and image features dominates the identification problem, and geometric computations integral to this process are simpler in 2D than in 3D. In particular, Beveridge et al (Beveridge, Weiss et al. 1989a; Beveridge, Weiss et al. 1989b) show that the determination of the optimal position of an object's 2D projection with respect to corresponding line features has an analytic solution in the two dimensions of image space. This closed form solution for line correspondence is a new result and we believe it to be a significant contribution. It is highly doubtful that the related 3D problem has an analytic solution for determining model positions that minimize point-to-line and point-to-plane distances.

Given that matches will seldom if ever be perfect, the emphasis must be on determining the 'best' of the imperfect matches. Hence matching is naturally posed in terms of optimization over the possible matches. By establishing an objective measure of match quality, the problem becomes one of determining the correspondence between model elements and data line segments for which the measure is optimal. The correspondence problem is combinatorial, and generally involves mapping one model line to many data lines. A second optimization problem is implicit in the correspondence problem. In order to measure the quality of a given model-data line correspondence, the best 2D position of the model with respect to the data must be determined, and the extent to which they do not spatially coincide must be measured. This we call the *fitting* problem. Hence, a match involves both model-data correspondence and an *associated best-fit* position.

The following is a sketch of the basic steps used to obtain a good model match:

- Determine the search space of correspondences. Lacking constraints on model position, all data lines segments possibly correspond to every model line segment. If constraints are available, only pairs of model and data lines satisfying these constraints need be considered.

- Determine promising model positions if the search space is large. Use these positions to determine constrained search subspaces made up of only correspondences consistent with the estimated position. A promising model position may be found either through a generalized hough transform or by identifying prominent features. The generalized hough technique involves an analysis of the space of possible two-dimensional spatial transforms to bring the model and data into alignment. Identifying a prominent feature may involve finding a distinctive part of a model, such as a corner and then using that to position the model as a whole.

- For each of the constrained search spaces obtained above, use iterative refinement to determine a best match. Upon each iteration perturb the correspondence, adding or deleting one or several data lines, and then determine the new best-fit model position and related match error. If the match error is reduced adopt the improved match. Stop when the match can no longer be improved. The best of the resulting matches is taken as the final match.

Results are presented for 2D model matching in Beveridge (Beveridge, Weiss et al. 1989b) using both synthetic data and images obtained from the robot vehicle. Sample results from this paper for one frame of a six frame image sequence is shown in Figures 5 and 6. The output from the 2D model matching system provide the input for the 3D pose refinement computation presented in the next section.

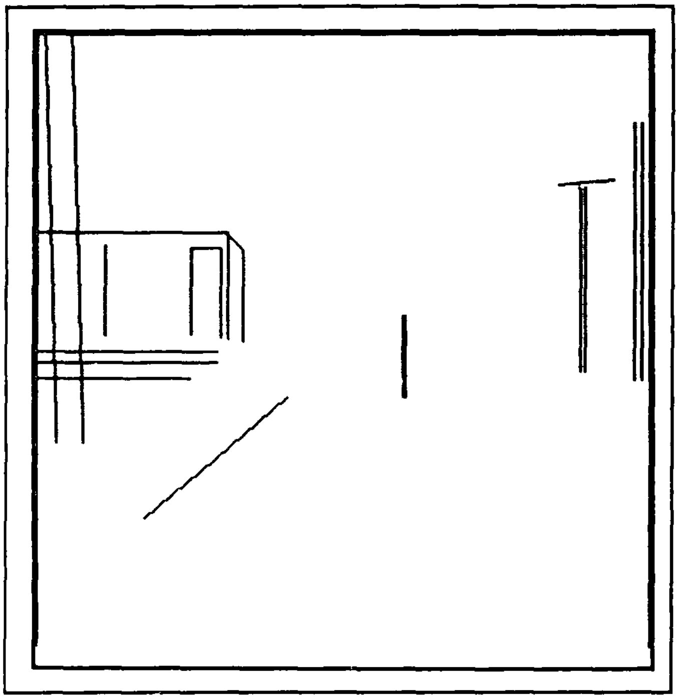


Figure 5 2D Modeling Matching Results. Projections of the six 3D navigation landmark models onto the 2D image plane using the current position of the robot.

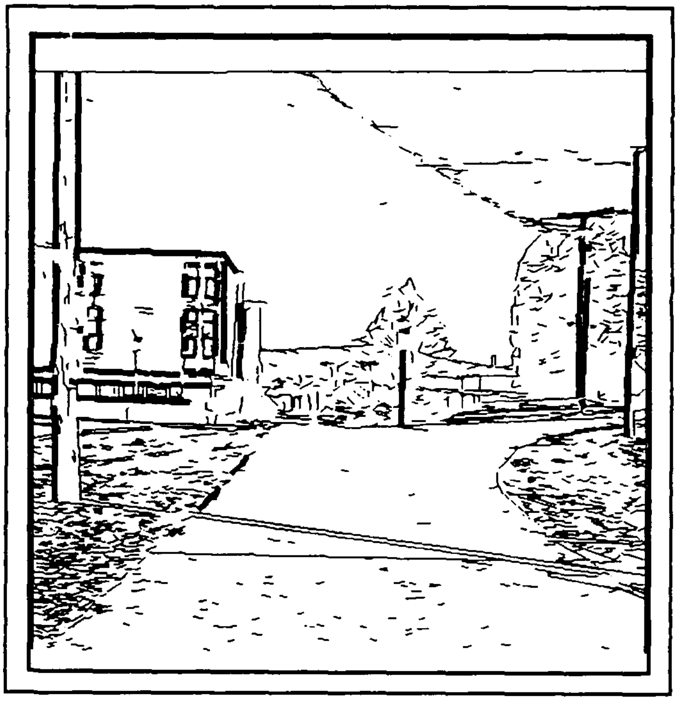


Figure 6. 2D Model Matching Results. Matches of model line segments with image line segments; the dark lines represent the matches. These matches are used by the 3D pose refinement module described in Section V.2

V.2 3D Pose Refinement

Kumar (Kumar 1989) develops a solution to and mathematical analysis of the pioblem of estimating camera location and orientation from a set of recognized landmarks appearing in the image. Given cor"espondences between the 3D landmark model lines and 2D image lines, the goal is to find the camera (or robot) rotation and translation which map the world coordinate system to the camera coordinate system under perspective projection. Because of the difficulties encountered in trying to establish accurate endpoint positions for lines (Kumar 1989; Lowe 1985; Williams and Hanson 1988), we assume that correspondences established between model and data are line correspondences and not endpoint correspondences. In addition, intrinsic camera parameters, such as focal length, field of view, center of the image, size of image, etc. are assumed to be known (Horn 1986; Kumar 1989; Lenz and Tsai 1988).

This problem, under various names and guises, has been addressed by several researchers, e.g. see (Ganapathy 1984; Horn 1987; Linnainmaa, D. et al. 1988; Wolf 1974); most of the techniques assume line endpoint data, are iterative in nature, and require an initial estimate. Liu, Huang, and Faugeras (Liu, Huang et al. 1988) present a solution to the "camera location determination" problem which works for both point and line data. Kumar's approach is based on their constraints, derived from the observation that the 3D lines in the camera coordinate system must lie on the projection plane formed by the corresponding image line and the optical center. Using this fact, Liu et. al. separated the constraints for rotation from those of translation, leading to a solution in which rotation is solved for first and then translation is obtained using the rotation results.

The technique developed by Kumar to solve for the rotation and translation parameters differs from that of Liu et al in two significant ways. First, rotation and translation are solved for simultaneously, which makes more effective use o the constraints and is more robust in the presence of noise. Second, the nonlinear technique used to solve for rotation and translation is adapted from Horn (Horn 1987) Kumar's version of this optimization technique provides much better convergence properties than does Liu et al's solution method based on Euler angles.

Kumar also develops uncertainty measures for the rotation and translation parameters. Noise in the data is assumed to be only in the image. The 3D model data is assumed accurate. The data for each image line can be specified by two parameters θ_i and ρ_i (a polar coordinate representation of lines). For the analysis, the noise for both θ_i and ρ_i is assumed to be Gaussian distributed with zero mean and known variances. Furthermore, the noise is assumed to be uncorrelated for different lines. Closed form expressions are developed for the variance of the error in the output parameters (rotation and translation) as a function of the input data and output translation and rotation values. Kumar shows that the error in the output parameters is linearly related to the noise in the input data. The reader can refer to Kumar's paper in these proceedings for more details.

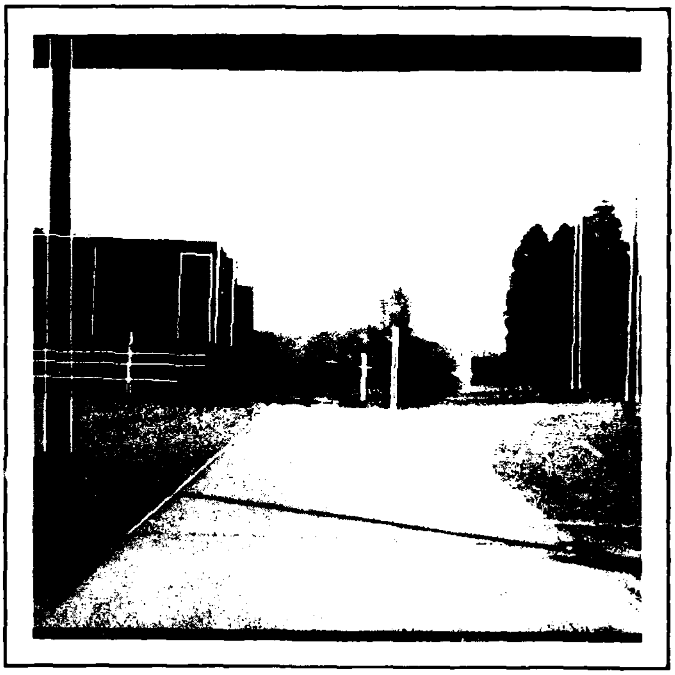


Figure 7. Results from 3D Pose Computation. The white lines are the 3D landmark segments reprojected onto the image plane after 3D pose refinement using the model line-data line matches shown in Figure 6.

Figure 7 shows the results for one frame *(the same frame shown in Figures 5 and 6)* of the six frame sequence used in one of the experiments. The figure shows the 3D model lines after projection back into the image plane using the vehicle "pose" computed by the 3D pose refinement algorithm which solves simultaneously for the rotation and translation parameters. For this particular frame, the errors (in feet) for the position of the robot (x,y,z) are (.1, 06, .03); additional results for the other frames of this sequence are given in Kumar's paper in these proceedings (Kumar 1989).

VI. CONCLUSIONS

The work presented here represents the current status of a long term research effort leading to the development of perceptually-based navigation systems for autonomous robots. The focus of the research is on environmental modeling, planning, plan monitoring, and vision. These four components are tightly coupled in a system which provide the flexibility and extensibility required for an experimental testbed for robot navigation.

Because the vagaries of the physical world affect plan execution in unknown ways, plans, no matter how carefully constructed, cannot simply be blindly executed. Each step of the plan must be carefully monitored and compared to expectations. The system accomplishes this by defining milestones associated with each planned action. The milestones act as preconditions for subsequent plan steps; the next step cannot be executed unless the milestone is satisfied. This assures a correspondence between the environmental model and the assumed position of the robot relative to the model and the actual position of the robot in the physical world. Failure to satisfy a milestone causes replanning to take place. Interweaving perception, planning, and action in this way makes specific what task is expected of perception and provides a means for focusing available knowledge on local goals.

Experimental results from the system thus far are encouraging, although a number of issues remain to be explored. Harvey's world is completely known, which is perhaps an unrealistic assumption for an autonomous robot. The perceptual servoing mechanisms assume that 3D landmarks can be accurately extracted from the geometric model of the environment. It remains to be seen how the requirement of complete knowledge can be relaxed yet still maintain the idea of perceptual servoing. Incorporating the type of reasoning demonstrated by the schema system (Draper, Brolio et al. 1989) might allow Harvey to respond to instructions like "....continue down North Pleasant street past the Graduate Research Center, then turn left and..."

A unique feature of the model matching component is the separation of the process of positional updating into two steps: 2D matching followed by 3D pose refinement. The robustness of this technique must be determined and its computational efficacy over many experiments in multiple domains must be explored.

Finally, navigation is an extremely computationally demanding task, yet real-time performance is crucial for a mobile automaton whose survival may depend upon reaching critical decisions in a short period of time. An ongoing aspect of the work reported here is the exploration of means by which the navigation task may be distributed over suitably configured parallel architectures. Two complementary lines of research are currently underway, utilizing a Sequent Symmetry multiprocessor system and the University of Massachusetts Image Understanding Architecture.(Weems, Levitan et al. 1989).

VII. References

Beveridge, J. R., J. Griffith, R. Kohler, A. Hanson and E. Riseman. (1989). "Segmenting Images Using Localized Histograms and Region Merging," IJCV, Vol. 2(3), pp. 311-347.

Beveridge, J. R., R. Weiss and E. Riseman. (1989a). "Matching and Fitting Sets of 2D Line Segments to Broken and Skewed Data," Dept. of Computer and Information Science, University of Massachusetts (Amherst), TR In preparation.

Beveridge, J. R., R. Weiss and E. Riseman. (1989b). "Optimizing Two-Dimensional Model Matching," Proc. of DARPA Image Understanding Workshop, Palo Alto, CA.

Brooks, R. (1986). "A Robust Layered Control System for a Mobile Robot," IEEE J. of Robots and Automation, Vol. RA-2(1), pp. 14-23.

Burns, J. B., A. Hanson and E. Riseman. (1986). "Extracting Straight Lines," IEEE PAMI, Vol. 8(4), pp. 425-455.

Connolly, C. (1989). "Geometer: Solid Modelling and Algebraic Manipulation," Dept. of Computer and Informtion Science, University of Massachusetts (Amherst), TR In Preparation.

Connolly, C. and R. Weiss. (1989). "Geometer: A System for Modelling and Algebraic Manipulation," Proc. of DARPA Image Understanding Workshop, Palo Alto, CA.

Dickmans, E. and V. Grafe. (1988a). "Applications of Dynamic Monocular Machine Vision," Machine Vision and Applications, Vol. 1, pp. 241-.

Dickmans, E. and V. Grafe. (1988b). "Dynamic Monocular Machine Vision," Machine Vision and Applications, Vol. 1, pp. 223-240.

Draper, B., J. Brolio, R. Collins, A. Hanson and E. Riseman. (1989). "The Schema System," IJCV, Vol. 2(3), pp. 209-250.

Eilenberg, S. and N. Steenrod. (1952). Foundations of Algebraic Topology, Princeton, NJ: Princeton University Press.

Fennema, C., A. Hanson, E. Riseman, R. Beveridge, R. Kumar, C. Connolly and R. Weiss. (1989). "Model Directed Mobile Robot Navigation," Dept. of Computer and Information Science, University of Massachusetts (Amherst), TR in preparation.

Fennema, C., E. Riseman and A. Hanson. (1988). "Planning With Perceptual Milestones to Control Uncertainty in Robot Navigation," Proc. of SPIE -- International Society for Photographic and Industrial Engineering, Cambridge, MA.

Fennema, C., E. Riseman and A. Hanson. (1989). "Planning with Perceptual Milestones to Control Uncertainty in Robot Navigation," Proc. of AAAI Spring Symposium Series (Working Notes: Robot Navigation), Stanford University, pp. 19-23.

Ganapathy, S. (1984). "Decomposition of transformation matrices for robot vision," Proc. of 1st IEEE Conference on Robotics, pp. 130-139.

Greenberg, M. and J. Harper. (1981). Algebraic Topology: A First Course, Reading, MA: Benjamin.

Horn, B. K. P. (1986). Robot Vision, Cambridge, MA: MIT Press.

Horn, B. K. P. (1987). "Closed-form solution of absolute orientation using unit quarternions," J. Opt. Soc. A., Vol. 4, pp. 629-642.

Kahn, P., L. Kitchen and E. Riseman. (1987). "Real-Time Feature Extraction: A Fast Line Finder for Vision-Guided Robot Navigation," Dept. of Computer and Information Science, University of Massachusetts (Amherst), TR 87-57.

Kanade, T., C. Thorpe and W. Whittaker. (1986). "Autonomous Land Vehicle Project at CMU," Proc. of ACM Computer Conference.

Kumar, R. (1989). "Determination of Camera Location and Orientation," Proc. of DARPA Image Understanding Workshop, Palo Alto, CA.

Lenz, R. and R. Tsai. (1988). "Techniques for calibration of the scale factor and image center for high accuracy 3-D machine vision metrology," IEEE-PAMI, Vol. 10(5), pp. 713-719.

Linnainmaa, S., H. D. and L. Davis. (1988). "Pose determination of a three-dimensional object using triangle pairs," IEEE-PAMI, Vol. 10(5), pp. 634-647.

Liu, Y., T. Huang and O. Faugeras. (1988). "Determination of Camera Location From 2D and 3D Line and Point Correspondences," Proc. of CVPR, Ann Arbor, MI, pp. 82-88.

Lowe, D. (1985). Perceptual Organization and Visual Recognition, Boston, MA: Kluwer Academic Publishers,.

Lowe, D. (1987). "The Viewpoint Consistency Constraint," IJCV, Vol. 1(1), pp. 58-72.

Lowrie, J., M. Thomas, K. Gremban and M. Turk. (1985). "The Autonomous Land Vehicle (ALV) Preliminary Road-Following Demonstration," Proc. of Intelligent Robots and Computer Vision (SPIE 579), D.P. Casasent, Ed., pp. 336-350.

Shafer, S., A. Stentz and C. Thorpe. (1986). "An Architecture for Sensor Fusion in a Mobile Robot," Proc. of IEEE International Conference on Robotics and Automation.

Toscani, G. and O. Faugeras. (1987). "Structure from Motion Using the Reconstruction and Reprojection Technique," Proc. of IEEE Workshop on Computer Vision, pp. 345-348.

Weems, C., S. Levitan, A. Hanson, E. Riseman, J. Nash and D. Shu. (1989). "The Image Understanding Architecture," IJCV, Vol. 2(3), pp. 251-282.

Williams, L. R. and A. R. Hanson. (1988). "Translating Optical Flow into Token Matches and Depth from Looming," Proc. of 2nd International Conference on Computer Vision, Tarpon Springs, FL, pp. 441-448.

Wolf, P. (1974). Elements of Photogrammetry, New York: McGraw Hill.

MATERIAL CLASSIFICATION AND SEPARATION OF REFLECTION COMPONENTS USING POLARIZATION/ RADIOMETRIC INFORMATION

Lawrence B. Wolff[1]
Computer Science Department
Columbia University
New York, N.Y. 10027

ABSTRACT

We present a unified approach to the problems of two distinct areas of computer vision; (i) classification of the intrinsic composition of material surfaces, and, (ii) separation of diffuse and specular reflection components at illuminated points on object surfaces. For the first area under consideration, the majority of object surfaces can be simply classified according to their basic electrical properties; *metal* objects (e.g., aluminum, copper) conduct electricity rather well while *dielectric* objects (e.g., rubber, plastic, ceramic) conduct electricity poorly. Classification of image regions according to whether they correspond to metal or dielectric material can provide important information both for scene understanding, e.g., it can be used to prune hypothesis trees, and for industrial inspection, e.g., it might be used in printed circuit board inspection where precise localization of dielectrics or metals are required. For the second area under consideration, a major hindrance to image understanding algorithms are the presence of specular highlights on object surfaces. Specular highlights appear on object surfaces where the specular component of reflection from illuminating light sources is so dominant that most detail of the object surface is obscured by a bright region of reflected light. By quantitatively separating diffuse and specular components of reflection intrinsic object detail can be restored in the diffuse component image. Also the diffuse and/or specular component images can be more readily used for vision algorithms that compute local surface normals from radiometric information.

Prior to this work no definitive vision algorithm to classify material composition is known to exist (other than some artificial encoding scheme). This excludes the very specialized work of applied physicists which is not practical for most vision applications. The technique presented for separation of reflection components extends the current limitations of previous methods to only dielectrics to include metal surfaces as well. All the techniques presented in this paper rely upon the empirical determination of the *polarization Fresnel ratio* . Thus the techniques for material classification and separation of reflection components are dependent upon the same experimental process, enabling them to be performed essentially in parallel. We show how once the polarization Fresnel ratio is computed at a pixel, how the object material at that pixel is classified as a dielectric/metal, and how the diffuse and specular components of reflection are obtained. Then two methods are presented which sl ›w how the polarization Fresnel ratio can be empirically determined at a pixel.

1 INTRODUCTION

Most object surfaces belong to one of two broad material classes: metals or dielectrics.[2] Metals have relatively low electrical resistivity and thus are good conductors. By definition, dielectric materials are electrical insulators. The difference in the conductive properties of metals and dielectrics in turn produces a difference in surface interaction with light which is electromagnetic radiation . The reflective characteristics of metals and dielectrics can be vastly different, especially with respect to polarization. It is this difference we will exploit to classify materials based on camera images.

The classification of surfaces as either metal or dielectric has many potentials in machine vision. First there are certain algorithms that assume particular material types (e.g., see [Klinker et al. 1988]). Secondly, there are numerous inspection tasks, e.g., printed circuit board inspection, where the objective is to determine/verify the placement of metals and insulators. It seems natural to directly compute the material composition rather than attempting to infer it from standard image observable (color/intensity) and local context. A third use of material classification is as an intrinsic property of surfaces to be used in image understanding algorithms. Its usefulness follows from the fact that it is invariant with almost all variations in imaging system (e.g., light source color, light source intensity, camera characteristics, etc.) and that many objects are composed of a mixture of metallic and dielectric subparts. Depending on the object database, the knowledge that certain components are metal/dielectric could significantly reduce the matching/pose determination time.

Other researchers, [Healey and Binford 1988], [Healey and Blanz 1988] have postulated that we can use the spectral content of reflect light, assuming we know the spectral output of the source, to determine material classification. While theoretically provoking, these papers do not propose an actual definitive algorithm to distinguish metals and dielectrics. It is not clear how much spectral resolution is required to classify a mate:ial surface. If 5 nm resolution is required, then an expensive monochromator will be needed to separate out the proper spectral components of reflected light.

We present a well defined technique for distinguishing dielectrics and metals based upon an analysis of the polarization properties of reflected light. Effectively, the degree to which a material surface polarizes light upon reflection gives information as to whether the surface is a metal or a dielectric. All that is required, besides an imaging camera, is a relatively inexpensive polarizing filter.

Another problem which we address in this paper is the separation of reflection components. Separation of diffuse and specular components of reflection from an object surface provides important information to image understanding

[1] This work was supported in part by ARPA grant #N00039-84-C-0165 and NSF grant IRI-88-00370. This work was supported in part by an IBM Graduate Fellowship Award.

[2] The two main exceptions to this statement are natural semiconductors (which are relatively uncommon), and coated surfaces where a substrate of one material class is covered with a transparent/translucent coating of the other.

algorithms. Specular highlight regions on an object surface not only obscure intrinsic detail but can easily deceive an image understanding algorithm into interpreting such a region as a separate object, or as a region on the same object with high albedo. The separation of shading and highlight components (i.e., diffuse and specular components) was first suggested in [Barrow and Tenenbaum 1978] as being useful for intensity analysis because each individual reflection component is more simply related to the illumination and viewer geometry than is the sum of the two reflection components together. This is particularly true for rough object surfaces where the specular component of reflection is expressed with respect to a microfacet distribution function which is a very complicated function of imaging geometry. Such microfacet distribution functions are presented in [Torrance and Sparrow 1967] and [Cook and Torrance 1981]. The complicated nature of specular reflection from rough surfaces makes it very difficult to implement methods such as photometric stereo, presented in [Woodham 1978], to determine local surface normals on smooth objects whose rough level of detail is within pixel resolution. Removal of the specular component of reflection from rough surfaces, leaving the diffuse component of reflection which is Lambertian in nature, makes implementation of photometric stereo feasible on these types of surfaces.

Presented in [Shafer 1985] is the *Dichromatic Reflection Model* for dielectric object surfaces which is used to separate diffuse and specular components of reflection based upon the color of the total reflected light. This model of reflection states that the color of light reflected from dielectric objects , represented as a vector in color space, is a linear combination of two color vectors; the color vector for the *body component*[3] (i.e., shading or matte reflection) and the color vector for the *interface component*[4] (i.e., highlight reflection). The color of the body component of reflection depends upon the dielectric makeup of the object material, whereas the color of the interface component is equal to the color of an illuminating light source. At specular highlights the color of the illuminant is added to the body color of the object. Experimental evidence is shown in [Klinker et al. 1988] that demonstrates the separation of the body and specular components of reflection into an image without specular highlights and an image of just the specular highlights. This technique would not work if the illuminator were the same color as the object.

A technique presented in this paper presents a new approach to the quantitative separation of diffuse and specular components of reflection on object surfaces. This approach is based on the polarization, rather than the spectral (i.e., color) properties of reflected light. The polarization state of light differs on specular highlights than on regions which have just diffuse reflection. This is due to the fact that the polarization state of the diffuse component is different from that of the specular component. This property is seen to have advantages over spectrally based techniques because it is universal to both metals and dielectrics and holds regardless of object or illumination color.

The techniques presented in this paper for classifying material surfaces and separating reflection components all depend upon the empirical determination of the same quantity at each image pixel, the *polarization Fresnel ratio*. The only difference between the techniques used to classify materials, and the techniques used to separate reflection components is how the polarization Fresnel ratio is interpreted or algebraically processed. After the polarization Fresnel ratio is determined, algorithms for classifying material surfaces as metal or dielectric, and for separating reflection components, can proceed in parallel.

The polarization Fresnel ratio is the ratio of the Fresnel reflection coefficients for perpendicular and parallel polarization, at a point on an object surface respective to some viewer orientation. We present two methods for determination of the polarization Fresnel ratio, both which use a polarizer mounted in front of the camera sensor. Two images are taken corresponding to designated perpendicular and parallel orientations of the polarizer. Both methods utilize the polarization/radiometric information from these two images.

2 BACKGROUND

We briefly present background information necessary for the discussion of the algorithms presented in this paper. We introduce basic information about polarization, imaging geometry, and the Fresnel reflection coefficients.

2.1 POLARIZATION AND IMAGING GEOMETRY

All light possesses a state of polarization which can be resolved into two independent component directions within the plane perpendicular to the direction of propagation. The magnitude of these two polarization components can be resolved by transmitting the light through a linear polarizer oriented in these respective directions. Light which is unpolarized will have equal transmittance through a linear polarizer regardless of its orientation. Such light is emitted by most typical lamps. Light which is partially polarized has a different transmittance for different orientations of a linear polarizer. Such light results from lamp light which is reflected off of material surfaces.

Polarization components are typically expressed with respect to the *specular plane of incidence* at a point on an object, which is the plane determined by the incident orientation of light rays from a point lighting element and the viewer orientation towards the object point. See figure 1. Note that the specular plane of incidence does not necessarily contain the surface normal. On rough surfaces specular reflection occurs off of microfacets which may not be oriented the same as the actual surface normal. The parallel polarization component lies within the specular plane of incidence, and the perpendicular polarization component is normal to the specular plane of incidence. When initially unpolarized light is specularly reflected off of a surface, the reflected light becomes slightly polarized towards the perpendicular polarization component. This can be observed as a slight increase in the transmitted radiance when light reflected from a surface is passed through a linear polarizer which is rotated from the parallel to the perpendicular orientation relative to the specular plane of incidence.

Referring again to figure 1, the *phase angle* is the angle between the incident orientation of the lighting element and the orientation of the camera sensor. The *specular angle of incidence* is equal to one-half the phase angle. The specular angle of incidence is equivalently the angle of incidence at which specular reflection takes place from the lighting element into the camera sensor.

[3] The body component of reflection is produced by light rays that penetrate into the surface of the object and then back out. The reason why the Dichromatic Reflection Model is not useful for metals is because they do not possess a body component of reflection.

[4] The interface component of reflection is produced by light rays that singularly or multiply reflect off of microfacets.

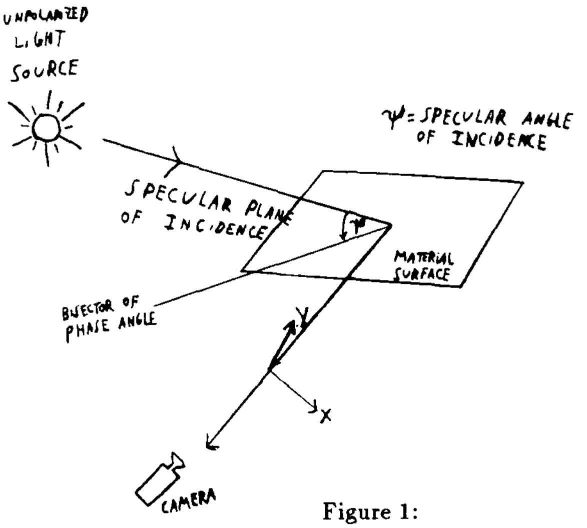


Figure 1:

2.2 THE FRESNEL REFLECTION COEFFICIENTS

The Fresnel reflection coefficients, $F_{\|}(\eta,\Psi)$, and $F_{\perp}(\eta,\Psi)$ describe the attenuation for parallel and perpendicular incident polarization light waves, respectively, that undergo pure specular reflection from a planar portion of material surface. In other words, if the incident radiance of the light wave is represented by N_i, the reflected radiance of the specularly reflected light wave is $N_i F_{\|}(\eta,\Psi)$ and $N_i F_{\perp}(\eta,\Psi)$, for parallel and perpendicular incident polarized light, respectively. The Fresnel reflection coefficients have a range from 0 to 1 inclusive, and are functions of index of refraction, η, and specular angle of incidence, Ψ.

In its most general form the index of refraction can be a complex number, represented as $\eta = n - i\kappa$. The term n is referred to as the ***simple*** index of refraction, and the term κ is called the ***coefficient of extinction.*** Dielectrics, which refers to the class of materials which do not conduct any electricity, have $\kappa = 0$, which means that they only possess a simple index of refraction n. For all dielectrics, $n > 1.0$. Most common plastics, glasses and ceramics have indices of refraction between 1.4 and 1.8. Some of the highest dielectric indices of refraction occur for precious gems (e.g. diamond) with $n > 2.7$. For all metals, $n, \kappa > 0$ and it is not all to unusual for n and κ to be higher than 10.0, especially for longer wavelengths of light.

In terms of Ψ and η, the Fresnel reflection coefficients are given by the following equations [Siegel and Howell 1981]:

$$F_{\perp}(\eta,\Psi) = \frac{a^2 + b^2 - 2a\cos\Psi + \cos^2\Psi}{a^2 + b^2 + 2a\cos\Psi + \cos^2\Psi}$$

$$F_{\|}(\eta,\Psi) = \frac{a^2 + b^2 - 2a\sin\Psi\tan\Psi + \sin^2\Psi\tan^2\Psi}{a^2 + b^2 + 2a\sin\Psi\tan\Psi + \sin^2\Psi\tan^2\Psi} F_{\perp}(\eta,\Psi)$$

where

$$2a^2 = [(n^2 - \kappa^2 - \sin^2\Psi)^2]^{1/2} + n^2 - \kappa^2 - \sin^2\Psi$$

$$2b^2 = [(n^2 - \kappa^2 - \sin^2\Psi)^2]^{1/2} - (n^2 - \kappa^2 - \sin^2\Psi) \ .$$

The Fresnel reflection coefficients are graphed in figures 2 and 3 for a dielectric and a metal respectively. Note that the value for $F_{\perp}$ is always greater than or equal to $F_{\|}$ for all specular angles of incidence ranging from 0^0 to 90^0.

Suppose that a light wave is incident on a material surface with a combination of parallel and perpendicular polarization components of magnitude $P_{\|}$ and $P_{\perp}$, respectively. Construct $0 \leq \alpha, \beta \leq 1$ such that $\alpha/\beta = P_{\|}/P_{\perp}$ and $\alpha + \beta = 1$. The corresponding Fresnel reflection coefficient is the linear combination $\alpha F_{\|}(\eta,\psi) + \beta F_{\perp}(\eta,\psi)$.

3 THE FRESNEL REFLECTANCE MODEL

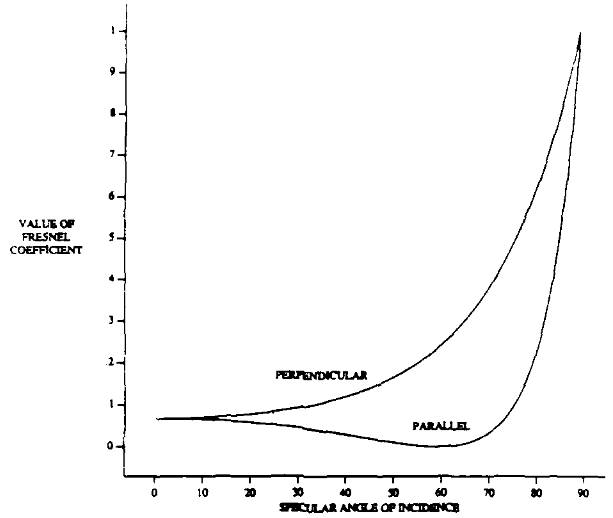
The techniques for material classification and separation of reflection components are based upon a simple, and yet, general reflectance model called the *Fresnel reflectance model.* Much of the motivation for this reflectance model is presented in [Wolff 1987].

Material surfaces are assumed to have a microscopic level of detail which consists of a statistically large distribution of specularly reflecting planar microfacets. Each planar microfacet is perfectly smooth. Most light that is reflected from a material surface arises from the following three phenomena:

- Light rays which specularly reflect off a planar microfacet a single time.
- Light rays which go through at least two multiple specular reflections amongst multiple planar microfacets.
- Light rays which penetrate into the top layer of the material surface and then are reflected back out.

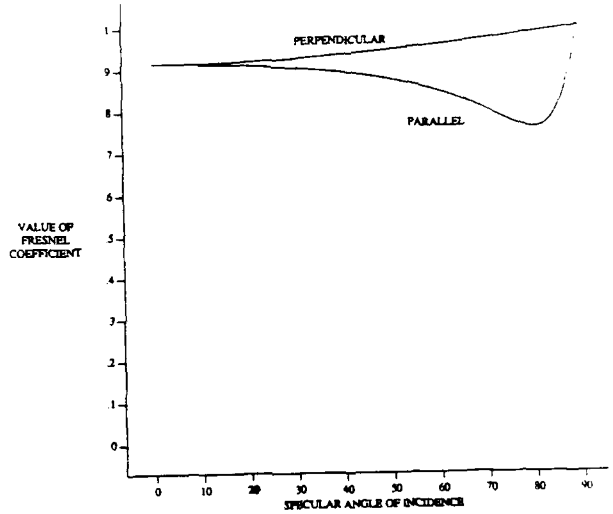
These three phenomena are illustrated in figure 4.

In the Dichromatic Reflection Model, the body component of reflection consists of the third phenomenon, while the interface component consists of the first two phenomena. In this paper the ***diffuse component*** of reflection will at least consist of the second and the third reflection phenomena. In addition to a microscopic planar microfacet level of detail,



FRESNEL REFLECTION CURVES FOR DIFFERENT INCIDENT POLARIZATION
DIELECTRIC N=1.7

Figure 2:



FRESNEL REFLECTION CURVES FOR DIFFERENT INCIDENT POLARIZATION
ALUMINUM METAL N=0.82, K=5.99 (at 546 NM)

Figure 3:

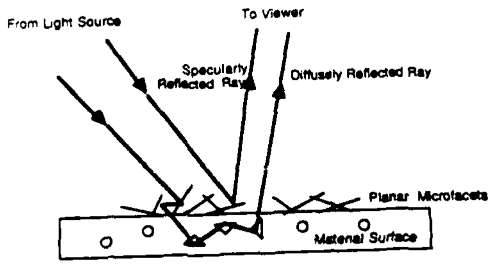


Figure 4:

some rough materials can also have a rough level of detail that is visible but yet smaller than a pixel projected out onto the object. This is true for brushed metal surfaces. Multiple reflections amongst this level of surface detail is considered to contribute to the diffuse component. Also protrusions from other regions of a surface material can contribute to the diffuse component of reflection at a point in the form of stray light reflections. The *specular component* will consist of only the first reflection phenomenon listed above.

The Fresnel reflectance model represents the diffuse and specular components of reflection respectively with the functions I_d and I_s. These are assumed to be functions of any number of physical parameters such as imaging geometry, surface roughness, wavelength, etc. . It is not necessary to know the exact form of these functions. The Fresnel reflectance model states, however, that the specular component function, I_s, is separable into $F(\eta, \Psi)L_s$ so that the reflected radiance for a given specular angle of incidence, Ψ, is represented by:

$$I_d + I_s = I_d + F(\eta, \Psi)L_s$$

where $F(\eta, \Psi) = \alpha F_{\|}(\eta, \Psi) + \beta F_{\perp}(\eta, \Psi)$ and some function L_s which is assumed to be a function of any number of physical parameters, except, incident polarization state and wavelength. That is, the dependence of the specular component of reflection upon incident polarization and wavelength is completely mediated by the Fresnel reflection coefficient.

Of most direct importance to the techniques presented in this paper is the description by the Fresnel reflectance model of the reflected polarization state of light. Suppose we are given incident light with polarization state (α, β) at a specular angle of incidence, Ψ, upon a material surface with index of refraction, η. The Fresnel reflectance model states that after reflection from the surface that the transmitted radiance through a polarizer oriented at angle θ with respect to the specular plane of incidence is

$$\frac{1}{2}I_d + \frac{\alpha F_{\|}(\eta, \Psi) cos^2\theta + \beta F_{\perp}(\eta, \Psi) sin^2\theta}{\alpha F_{\|}(\eta, \Psi) + \beta F_{\perp}(\eta, \Psi)} I_s \,. \tag{1}$$

For both metals and dielectrics the diffuse component of reflection arises from multiple random reflection processes. Because of this, the polarization of the diffuse component of reflection is assumed to be always nearly completely unpolarized. This is why there is a constant coefficient $\frac{1}{2}$ in front of I_d for all polarizer orientations, θ.

For incident unpolarized light ($\alpha = \beta = 1/2$), typically emitted from most light sources, the transmitted radiance, $k_{\perp}, k_{\|}$ of reflected light through a polarizer oriented perpendicular and parallel, respectively, to the specular plane of incidence, is given by

$$\frac{1}{2}I_d + \frac{F_{\perp}}{F_{\|} + F_{\perp}} I_s = k_{\perp} \tag{2}$$

$$\frac{1}{2}I_d + \frac{F_{\|}}{F_{\|} + F_{\perp}} I_s = k_{\|} \tag{3}$$

where functional variables have been supressed.

The Torrance-Sparrow reflectance model [Torrance and Sparrow 1967] is a specific instantiation of the Fresnel reflectance model. It assumes specific functions of surface roughness and imaging geometry. The vision techniques presented in this paper only need assume the Fresnel reflectance model in its most general form, without worrying about the specific nature of reflectance functions beyond the Fresnel reflection coefficients.

4 USING THE POLARIZATION FRESNEL RATIO TO CLASSIFY MATERIALS AND SEPARATE REFLECTION COMPONENTS

Given that the polarization Fresnel ratio, $q = \frac{F_{\perp}}{F_{\|}}$, has been computed at each pixel corresponding to illuminated object points, we describe how material classification is peformed and how reflection components are separated. In the next section we describe methods for how the polarization Fresnel ratio is computed.

4.1 MATERIAL CLASSIFICATION

Figure 5 shows the polarization Fresnel ratio curves as a function of specular angle of incidence corresponding to the Fresnel curves in figures 2 and 3. Since the respective polarization Fresnel ratio curves in figure 5 are typical for dielectrics and metals, there appears to be a definitive relationship between the magnitude of the polarization Fresnel ratio and dielectric and metallic materials. In particular, for a fairly large range of specular angles of incidence the polarization Fresnel ratio for dielectrics is much larger than for metals. Consider the threshold boundaries for the polarization Fresnel ratios at 2.0 and 3.0 in figure 5 (represented as horizontal dashed lines). Within the specular angles of incidence from 35^0 to 80^0 (phase angles from 70^0 to 160^0), a pixel corresponds to dielectric material if and only if the polarization Fresnel ratio is greater than or equal to 3.0. If within this same range of specular angles of incidence the polarization Fresnel ratio is less than or equal to 2.0, the pixel corresponds to metallic material.

Recall that the polarization Fresnel ratio for the dielectric in figure 5 corresponds to the Fresnel curves in figure 2 with index of refraction n=1.7 . What about including all dielectrics having index of refraction between 1.0 and 2.0 ? To preserve the property that dielectrics with $1.0 < n \leq 2.0$ have polarization Fresnel ratio greater than or equal to 3.0, the range of specular angles of incidence needs to be slightly more restricted to be between 40^0 and 70^0 (phase angles from 80^0 to 140^0). This includes practically all commonly occurring dielectrics except rare gems. To include such rare dielectrics a further restriction to specular angles of incidence ranging from 45^0 to 65^0 (phase angles from 90^0 to 130^0) is necessary.

In examining the optical tables in [Physics Handbook], we have found that the polarization Fresnel ratio for metals never exceeds 1.5 for all specular angles of incidence. In actual experimentation we have found that some commonly occuring metals can have maximum polarization Fresnel ratios about 1.8 or 1.9 . This is probably due to nonhomogeneities in the metal and/or oxidation. Picking polarization Fresnel ratio thresholds of 2.0 and 3.0 allows for robustness of classification of metals and dielectrics in the presence of usual camera sensor errors. These thresholds are not heuristic, but arise directly from the physics of Fresnel reflection coefficients.

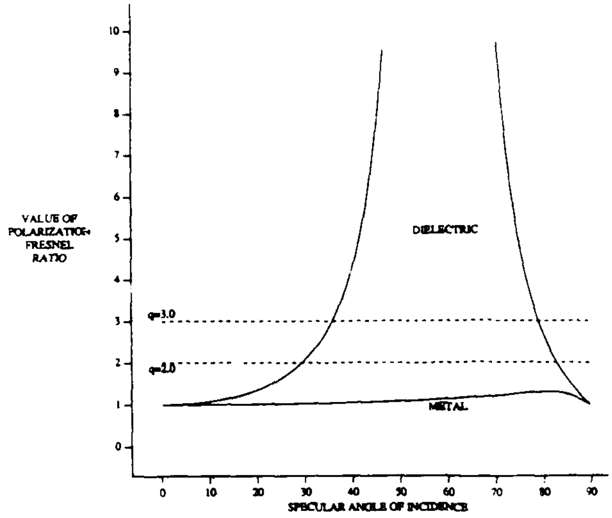


Figure 5:

4.2 SEPARATION OF REFLECTION COMPONENTS

Consider equations 2 and 3 of the Fresnel reflectance model equivalently expressed in terms of the polarization Fresnel ratio, $q = \frac{F_\perp}{F_\parallel}$:

$$\frac{1}{2}I_d + \frac{q}{1+q}I_s = k_\perp \tag{4}$$

$$\frac{1}{2}I_d + \frac{1}{1+q}I_s = k_\parallel\ . \tag{5}$$

Clearly a determination of q at each pixel provides simultaneous linear equations in which to solve for I_d and I_s at each pixel. The larger q is, the more accurate the solution of I_d and I_s are with respect to given errors in $k_\perp$ and $k_\parallel$.

5 EMPIRICAL DETERMINATION OF THE POLARIZATION FRESNEL RATIO

We propose two methods for determining the polarization Fresnel ratio from specularly reflected light. The first method approximates the polarization Fresnel ratio at each pixel from the ratio of the perpendicular and parallel polarization components of the total reflected light. This method works well for material classification at object points where the specular albedo is much larger than the diffuse albedo. Unfortunately, because of the nature of the approximation used, the first method is not suitable for separation of reflection components. The second method is a global technique which utilizes polarization information from many pixels within a specular region to establish the polarization Fresnel ratio for that region. This works well for both material classifiaction and separation of reflection components. However, because it is a global method, problems may arise if the scene is composed of many different materials within a small region of the image.

5.1 APPROXIMATING THE POLARIZATION FRESNEL RATIO BY USING THE RATIO OF POLARIZATION COMPONENTS

From equations 4 and 5 above it is easy to derive

$$q = \frac{k_\perp - \frac{1}{2}I_d}{k_\parallel - \frac{1}{2}I_d}\ . \tag{6}$$

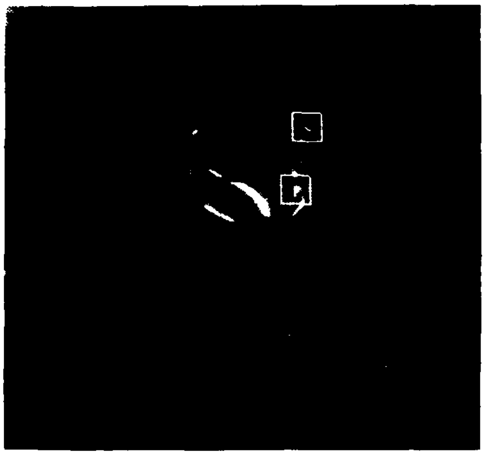
Suppose, as is true for most surfaces, that the specular albedo is much stronger than the diffuse albedo. For such surfaces, $I_d \ll I_s$ for points on specular highlight regions. Observe again equations 4 and 5. Since $F_\perp \geq F_\parallel$, we have that $k_\perp > (1/2)I_s$ and therefore for points on specular regions $I_d \ll k_\perp$. If in addition $I_d \ll k_\parallel$, then from equation 6:

$$\frac{F_\perp}{F_\parallel} \approx \frac{k_\perp}{k_\parallel}\ . \tag{7}$$

That is, the ratio of the observed image irradiance values at a pixel corresponding to a point on a specular region for perpendicular and parallel orientations of a polarizing lens in front of the camera, is a good approximation to the polarization Fresnel ratio. The condition $I_d \ll k_\parallel$ is true for specular regions on metals and for specular regions on dielectrics assuming a specular angle of incidence far from the Brewster angle. As can be seen from equation 6 and simple arithmetic, the ratio $k_\perp/k_\parallel$ always theoretically underestimates the true polarization Fresnel ratio. In the case of a dielectric where the specular angle of incidence is close to the Brewster angle

$$\frac{F_\perp}{F_\parallel} \gg \frac{k_\perp}{k_\parallel}\ .$$

For dielectrics using a specular angle of incidence equal to the Brewster angle, $k_\perp/k_\parallel$ will be extremely large ($\gg 3$), underestimating an infinite Fresnel ratio.



PIXEL VALUES AT SPECULAR HIGHLIGHTS FOR TEAPOT WITH HANDLE

HIGHLIGHT ON BASE OF TEAPOT

PERPENDICULAR ORIENTATION				PARALLEL ORIENTATION				Kperp/Kpara			
70	129	110	71	48	93	78	38	1.4	1.6	1.7	1.7
84	179	222	167	56	127	165	112	1.7	1.7	1.9	1.8
145	237	233	225	128	234	235	209	1.2	1.0	<1	1.4
91	189	208	121	54	141	167	92	1.9	1.7	1.4	1.5

HIGHLIGHT ON HANDLE OF TEAPOT

PERPENDICULAR ORIENTATION				PARALLEL ORIENTATION				Kperp/Kpara			
6	8	60	129	3	6	10	6	-	-	3.6	>10
13	126	191	78	3	5	3	4	-	>10	>10	>10
82	93	67	55	4	4	3	3	>10	>10	>10	>10
25	27	24	6	3	3	3	4	-	-	-	-

Figure 6: Approximating the polarization Fresnel ratio on specular highlights on a metal teapot with plastic handle.

Approximating the polarization Fresnel ratio using equation 7 appears from initial experimentation to be very good for classifying materials as metal or dielectric. Figure 6 from [Wolff 88a] shows the classification of specular highlights on a metal teapot with plastic handle, illuminated with a small lamp light. (The picture was taken with extra room lighting, but the specular highlights marked by squares are from the small lamp light.) Recall that, assuming the phase angle is between 80^0 and 140^0, for a metal the polarization Fresnel ratio is less than 2.0, while for a dielectric the polarization Fresnel ratio is greater than 3.0 . The approximation to the polarization Fresnel ratio is shown pixelwise in a 4x4 pixel neighborhood of the center of each specular highlight. Note that if $k_\perp$ is less than gray value 50 that no ratio is computed because then it does not lie on a specular highlight. The value of the approximation to the polarization Fresnel ratio is consistent with whether the specular highlight falls on a metal or dielectric. This is true also for figure 7 which shows specular highlights on various objects.

So far we have shown experimental results using a single point light source resulting in essentially only a single specular plane of incidence at points on a specular highlight region. If we were to use an extended light source we increase the number of dominant specular points which we can classify. However the specular plane of incidence now varies amongst specular points. Also we must insure that the entire extent of the extended light source is such that all lighting elements of the source form a phase angle with the viewer orientation within the specified range for robust material classification.

According to the Fresnel reflectance model we can isolate the specular plane of incidence at a pixel by observing extrema in transmitted radiance through the polarizer with respect to polarizer orientation. Referring back to expression 1, since always $F_\| \leq F_\perp$, and assuming an unpolarized light source ($\alpha = \beta = 1/2$), a transmission minimum occurs when $\theta = 0$ (parallel to the specular plane of incidence) and a transmission maximum occurs when $\theta = 90^0$ (perpendicular to the specular plane of incidence). In general, we can measure the transmitted radiance through the polarizer, at a pixel, for incremental rotations of the polarizer. After finding out the orientation of the specular plane of incidence, we can compute q from transmitted radiance, k_α, k_β for any α, β relative to the specular plane of incidence. Again using expression 1 we can derive that

$$q = \frac{k_\beta sin^2\alpha - k_\alpha sin^2\beta}{k_\alpha cos^2\beta - k_\beta cos^2\alpha} .$$

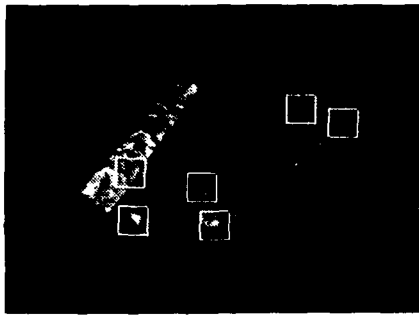
For better estimation of q we average all the q values obtained from the transmitted radiance values taken for all incremental rotations of the polarizer.

Figure 8 shows an extended light source illuminating a printed circuit board. The extended light source used was a standard light table (used in photographic work) approximately 60cm by 90 cm. The light source itself is a 60 watt fluorescent tube, surrounded with reflecting material directing the light toward a diffusing plate. The intensity is not, however, uniform over the extend light source surface.

The camera images were acquired using our PIPE machine, and a Video logic camera. To minimize the effects of camera error (and some of the problems inherent in digitizing images when using a fluorescent light source), we averaged the scene over 64 frames.

In figures 8a and 8b we see the same scene with the polarizer oriented perpendicular and parallel to the vertical specular plane of incidence. The objects in the scene consist of a PC board on the right, a sheet of blue acetate on the left with a metal wrench placed just above the acetate. The board/acetate were oriented in front of the camera such that the phase angles were approximately in the range required by the theory, with the front of the board at approximately 75°.

For each scene, we acquired 9 images at 10 degree intervals. The algorithm then used (pixelwise) the maximum of the transmitted light through the polarizer to define the specular angle of incidence for each pixel. Averaging the 36 values of q obtained (disregarding some to avoid numerical difficulties) we were able to obtain the q image shown in 8c. The q values were scaled up in gray value representation to visually see their relative magnitude across the scene. Note the black "tag" on the acetate in figure 8c. This tag is made of paper with high albedo and is not a metal. The high diffuse component washed out any specular component here. The completely white portions in figure 8c are points we could not process (mostly from shadow regions in the original scenes). One can see that the circular regions (corresponding to bolts or large solder connections) are considerably darker than most of the "lines" of metal. This is because most of the "lines" are actually metal coated with a dielectric plastic. In figure 8d we see the q image threshold with a value of 1.7. All the



SPECULAR HIGHLIGHT ON COPPER PENNY

PERPENDICULAR ORIENTATION				PARALLEL ORIENTATION				Kperp/Kpara			
25	93	66	30	18	79	71	26	-	1.3	-	-
31	27	31	28	20	21	25	24	-	-	-	-
25	29	25	25	17	22	19	18	-	-	-	-
20	25	26	26	13	18	21	19	-	-	-	-

SPECULAR HIGHLIGHT ON CERAMIC CUP

PERPENDICULAR ORIENTATION				PARALLEL ORIENTATION				Kperp/Kpara			
8	8	7	7	7	7	6	7	-	-	-	-
11	174	117	15	10	16	17	9	-	6.1	6.7	-
4	9	9	12	7	8	8	11	-	-	-	-
9	9	8	8	8	8	7	7	-	-	-	-

SPECULAR HIGHLIGHT ON WATER

PERPENDICULAR ORIENTATION				PARALLEL ORIENTATION				Kperp/Kpara			
13	9	9	8	12	9	8	8	-	-	-	-
46	29	15	9	25	17	11	9	-	-	-	-
9	47	30	39	10	22	30	22	-	-	3.1	-
7	7	7	8	7	6	7	8	-	-	-	-

SPECULAR HIGHLIGHT ON PLASTIC

PERPENDICULAR ORIENTATION				PARALLEL ORIENTATION				Kperp/Kpara			
75	195	234	180	41	88	155	88	1.8	3.4	3.1	3.0
150	225	232	177	61	105	146	83	3.7	4.1	3.2	3.1
134	215	210	137	39	81	86	44	4.4	4.9	3.9	4.2
127	105	102	84	27	27	26	20	4.5	4.4	4.2	3.6

SPECULAR HIGHLIGHT ON ALUMINUM FOIL

PERPENDICULAR ORIENTATION				PARALLEL ORIENTATION				Kperp/Kpara			
115	144	112	60	102	140	118	61	1.2	1.0	<1	<1
138	160	99	58	132	163	114	56	1.1	<1	<1	1.0
101	98	56	97	97	108	59	87	1.1	<1	<1	1.2
97	90	71	113	91	97	68	103	1.1	<1	1.1	1.1

SPECULAR HIGHLIGHT ON QUARTER

PERPENDICULAR ORIENTATION				PARALLEL ORIENTATION				Kperp/Kpara			
50	53	52	57	38	40	38	42	1.2	1.3	1.3	1.3
101	131	73	59	88	103	58	39	1.2	1.4	1.3	1.4
73	158	192	70	67	138	186	82	1.2	1.3	1.1	<1
55	90	101	50	43	69	99	44	1.2	1.5	1.0	1.0

Figure 7: Approximating the polarization Fresnel ratio on specular highlights on an assortment of metals and dielectrics.

black regions are below that threshold, and are pure metal regions (remember we underestimate q.) Figure 8e shows the same scene thresholded with a value of 2.5, and the coated materials show up as quasi-metals.

If the approximation of the polarization Fresnel ratio in 7 does a good job for material classification, what about using this approximation for separating reflection components, and computing specular angle of incidence for surface normals ? The problem with separating out reflection components under the same lighting conditions for which $k_\perp/k_\parallel$ is taken as the true polarization Fresnel ratio is that I_d is assumed to be zero by virtue of this approximation. However it is conceivable that after computing this approximation for an object scene illuminated using an extended light source so as to "calibrate" the polarization Fresnel ratio, that this approximation may be useful for the separation of reflection components under different lighting conditions. There will potentially be problems at points on dielectrics with specular angle of incidence near the Brewster angle. Otherwise, it is expected that this approximation will work well for computing specular angle of incidence for surface normals. Experimentation will be performed to verify this.

5.2 A GLOBAL METHOD FOR DETERMINING THE POLARIZATION FRESNEL RATIO

In [Wolff 88b] a global method is presented to determine the polarization Fresnel ratio on specular highlight regions of objects illuminated by a point light source. A point light source is used so that specular reflection from various object points occur from light rays which have approximately the same incident orientation. Using a point light source, all points in a relatively small specular highlight region on the same object material surface should therefore have approximately the same polarization Fresnel ratio. By a relatively small specular highlight region, we mean a specular region small enough so that all specular reflection into the camera sensor occurs along approximately the same specular plane of incidence.

From equation 6 we easily arrive at the equation:

$$k_\perp = qk_\parallel + \frac{1-q}{2}I_d \, . \tag{8}$$

We can view equation 8 as a linear equation in variables $(k_\parallel, k_\perp)$ with slope q and $k_\perp$-intercept equal to $\frac{1-q}{2}I_d$. Recall again that $k_\perp$ and $k_\parallel$ are the transmitted radiance through the polarizer, in front of the camera sensor, oriented perpendicular and parallel to the specular plane of incidence, respectively. Equation 8 implies that a conclusion from the Fresnel reflectance model is that if the 2-tuple $(k_\parallel, k_\perp)$ varies from point to point on an object specular region, that it must be constrained along the linear locus which is specified. As a result of plotting individual 2-tuples $(k_\parallel, k_\perp)$ for each pixel corresponding to a part of the specular region, a linear locus of points will arise. The slope of this linear locus is the value of the polarization Fresnel ratio.

The 2-tuple $(k_\parallel, k_\perp)$ can vary from point to point on an object specular region for at least two reasons:

- The light source, however much it approximates a point source, is still extended. The spatial inhomogeneity of the light source with respect to emitted intensity produces specularly reflecting rays of varying intensity from point to point on a specular region of an object surface.
- The object surface itself can be rough and even if the light source is perfectly spatial homogeneous with respect to emitted intensity (which it never is) the variation in the per unit area of rough surface detail that specularly reflects

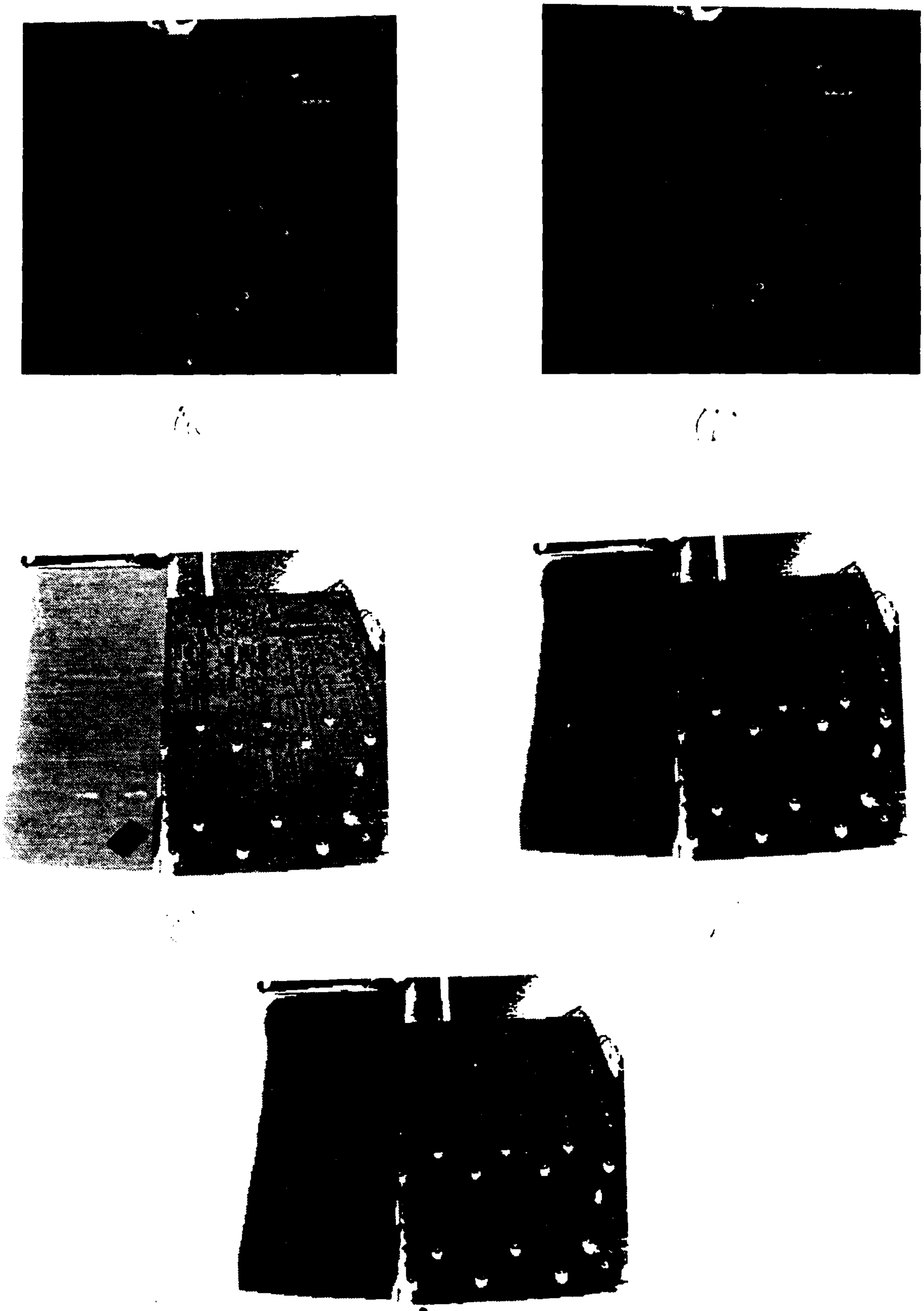


Figure 8:

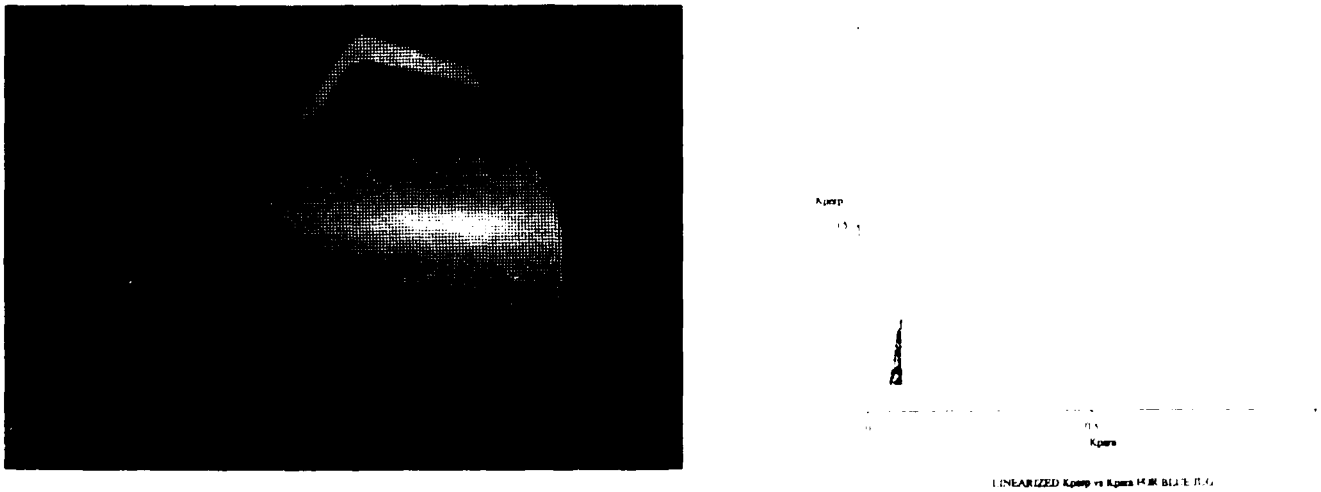


Figure 9: Plastic jug illuminated by small lamp light and associated polarization plot cluster.

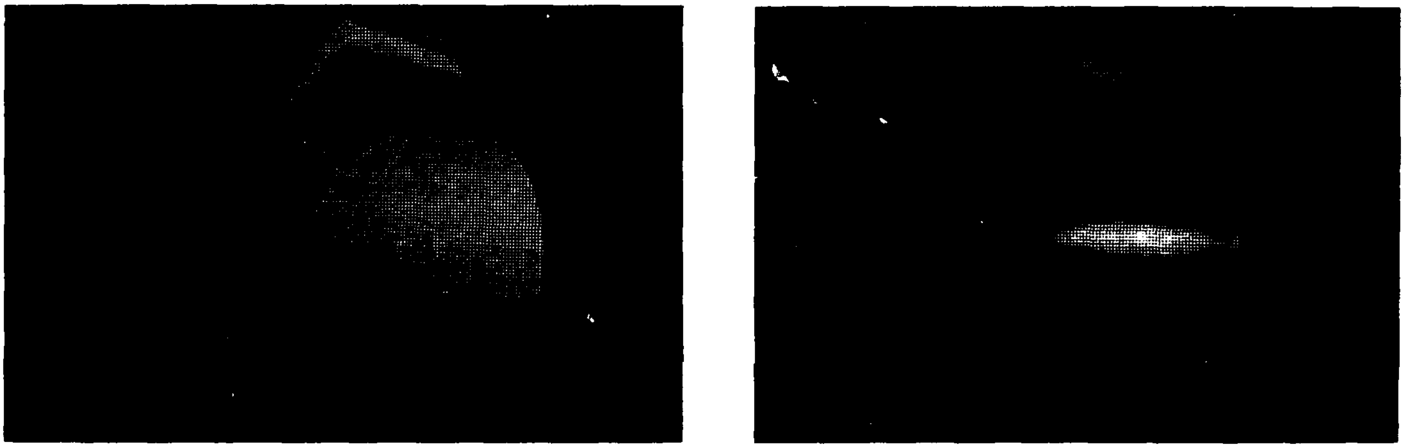


Figure 10: Separated diffuse and specular reflection components for plastic jug.

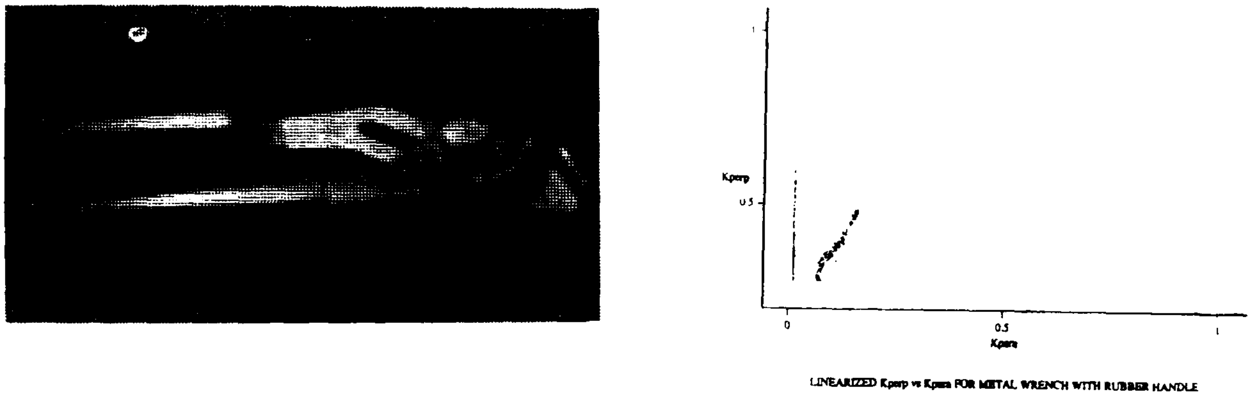
light into the camera sensor (e.g. microfacets) from point to point in turn produces variations in the magnitude of the specular component of reflection.

Figure 9 is taken from [Wolff 88b] and shows a picture of a plastic jug with specular region produced from a small lamp light source. Next to the picture of the plastic jug is a plot of polarization 2-tuples from pixels with $k_\perp$ above some background noise gray level. The values for $(k_\parallel, k_\perp)$ are radiometrically linearized to be within a range from 0 to 1. Practically all the points depicted in the polarization 2-tuple plot correspond to object points within the specular region of the plastic jug. The polarization 2-tuples form what appears to be a tapered linear cluster. The thickening of the linear cluster at the base illustrates a deviation from a pure line caused by a varying diffuse component, I_d, across the specular region. *Not all points within the specular region on the plastic jug have the same local surface orientation, and* therefore they do not have the same diffuse component, I_d . Equation 8 shows that if I_d varies slightly for some of the polarization 2 tuples, that these points lie on linear clusters which are rigid translations so that their $k_\perp$ intercepts are consistent. Thus the apparent thickening of the linear cluster. The thickening occurs mostly at the base of the cluster because these represent pixels on the periphery of the specular region where the specular component is not as strong.

A least squares linear fit to the polarization plot in figure 9 gives a slope of $q \approx 13.2$. This value is used for the polarization Fresnel ratio and then subsequently used in equations 4 and 5 to resolve I_d and I_s. The results are the images in figure 10. Note that in addition to separating out reflection components we have simultaneously classified the material surface as a dielectric ($q \gtrsim 3.0$) assuming that the material within the specular region is representative of the entire jug. The lamp light source is within the specified phase angle range for robust material classification.

Figure 11 shows the image of a metal wrench with rubber handle illuminated again by a smali lamp light. The polarization plot now shows two distinct linear clusters pointing out that there are at !east two very different materials contained in the image. The nearly vertical linear cluster has a slope of 151 whereas the other linear cluster was evaluated with a least squares linear fit to have slope 1.43 . Polarization 2-tuples for pixels corresponding to the specular region on the rubber handle were verified to lie on the linear cluster with slope 151, whereas polarization 2 tuples for pixels corresponding to the metal region were found to lie on the linear cluster with ope 1.43 . Hence we have a correct segmentation of dielectric and metal specular regions.

Note that the polarization cluster for the metal specular region is a bit spread out. This results from the presence of different metalic materials within the metal specular region. *In particular the metal for the bolts is different from the* metal of the body of the wrench. Also the metal of the body of the wrench, and the bolts that hold it together are most



LINEARIZED Kperp vs Kpara FOR METAL WRENCH WITH RUBBER HANDLE

Figure 11: Metal wrench with rubber handle illuminated by small lamp light and associated polarization plot cluster.

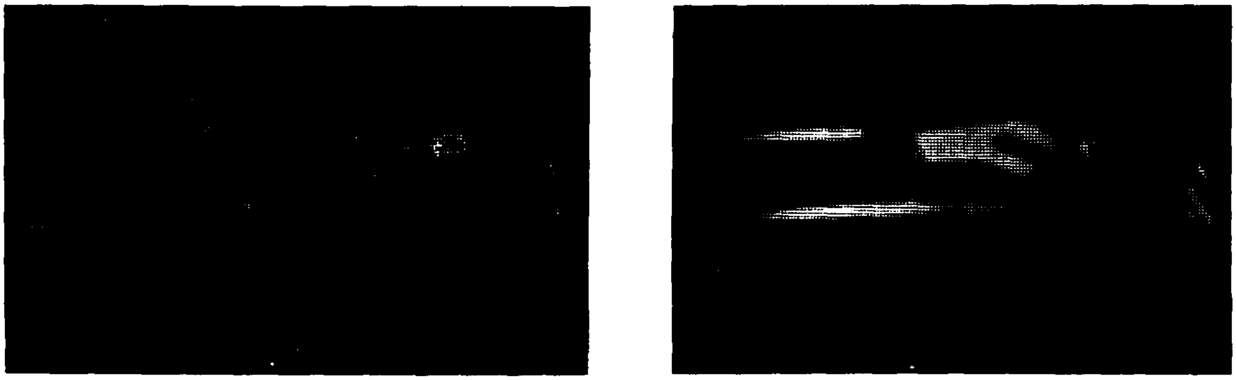


Figure 12: Separated diffuse and specular reflection components for metal wrench with rubber handle.

likely inhomogeneous. Thus the cluster for the metal is actually a mixture of different q values. The specular region on the rubber handle lies on a flat area of almost constant local surface orientation at each point. This accounts for the ideal linear shape of the polarization cluster for the specular region on the rubber handle.

Figure 12 shows the separation of diffuse and specular reflection components. The value of the polarization Fresnel ratio, q, used to separate reflection components at a pixel within a specular region corresponds to the cluster which the corresponding polarization 2-tuple for the pixel, lies.

In figure 13 there is a picture of a cup on which there is a specular region which crosses between two different dielectric paint regions. The polarization plot for the specular region in this picture shows a major problem inherent to determining the polarization Fresnel ratio from a global clustering technique. Within this polarization cluster there are two distinct linear polarization clusters that cannot be resolved. The slope of each of the two unresolved linear polarization clusters correspond to the polarization Fresnel ratio for each of the two dielectric regions. The slope of the entire cluster as a whole is an unknown mixture of two slope values. The separation of diffuse and specular reflection components at each pixel shown in 14 used one of 2 polarization Fresnel ratio values (one for each dielectric paint region) obtained by manual geuss work. An initial estimate of the polarization Fresnel ratios was provided by the slope of the entire polarization cluster.

We have seen that determination of the polarization Fresnel ratio from the slope of linear polarization plots does a fairly nice job of separating diffuse and specular reflection components, as well as classifying material surfaces. Unfortunately this global method works well on scenes with only a few objects and when the illuminating light source is small producing specular regions from incident rays with approximately the same orientation. The problem with generalizing this method to extended light sources is the inability to cluster up polarization 2-tuples for pixels corresponding to object points with the same polarization Fresnel ratio. We can segregate polarization 2-tuples for pixels according to the same specular plane of incidence, but still cannot guarantee that the specular angle of incidence is the same for polarization 2-tuples in the cluster. We can however use extended linear light sources which can be specially made to guarantee that if on a specular region if two pixels have the same specular plane of incidence, then the specular angle of incidence is approximately the same. As shown in figure 13, even for simple scenes there can exist severe problems in resolving polarization clusters if there exist different materials with similar polarization Fresnel ratios. These types of pattern recognition problems with resolving polarization clusters is seen to be the major disadvantage of this technique.

A very important aspect of producing polarization clusters which are linear is to corroborate the Fresnel reflectance

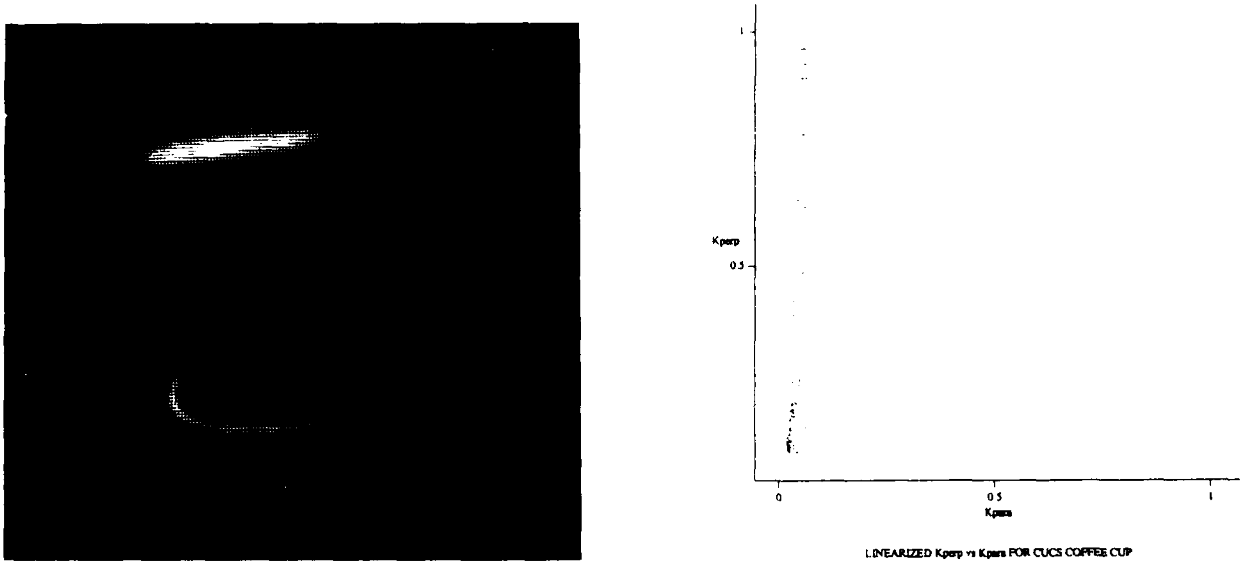


Figure 13: Painted cup illuminated by small lamp light and associated polarization plot cluster.

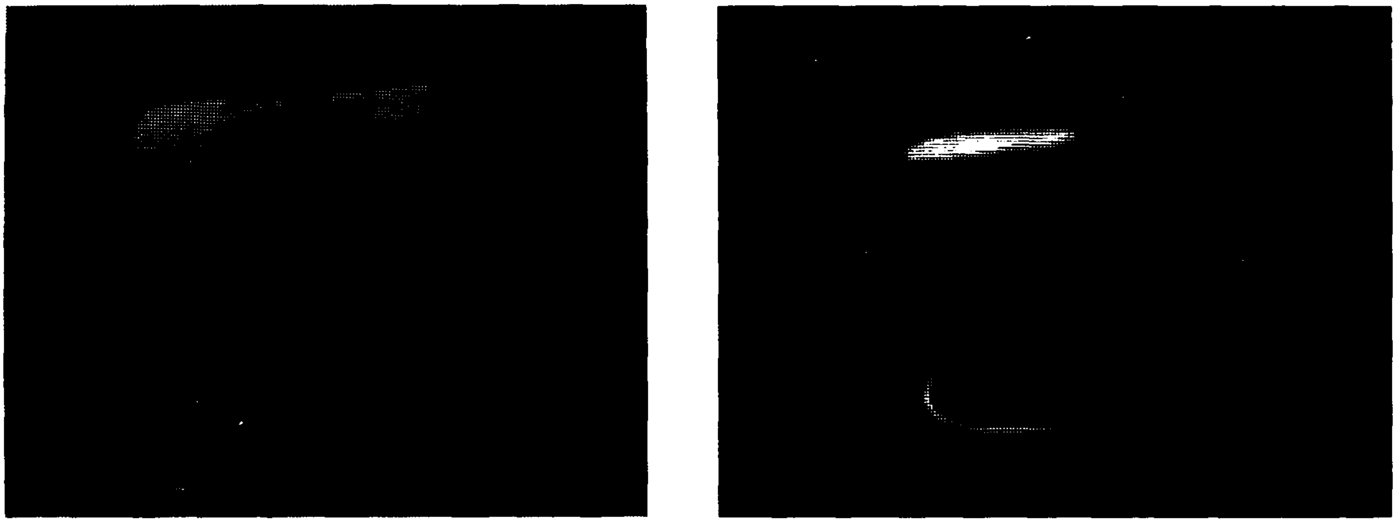


Figure 14: Separated diffuse and specular reflection components for painted cup.

model which predicts this according to equation 8 in the first place.

6 CONCLUSION AND FUTURE WORK

We have demonstrated vision techniques that (i) classify the composition of material surfaces as being dielectric/metal, and, (ii) separate diffuse and specular reflection components at illuminated object points. The application of polarization as well as radiometric information to these two areas of computer vision is not only novel but as was shown provides a definitive unified methodology for simultaneously classifying material surfaces and separating reflection components. Previously no known material classification vision techniques existed and existing methods to separate reflection components were limited to dielectric surfaces. The global method for determining the polarization Fresnel ratio was seen to have potential problems in the presence of different, but very similar material composition in the same scene. We hope to circumvent such problems by experimenting with other methods for determining the polarization Fresnel ratio that are local, only requiring computation at a single pixel.

The theory has already been drawn up for unification of these techniques with still a third area of computer vision; determination of local surface normals on a smooth surface. If the bulk of specular reflection from an object point occurs into the camera sensor through a single specular plane of incidence then the Fresnel reflectance model implies that this plane can be determined using a polarizer in front of the camera. Assuming that this specular plane of incidence contains the surface normal (true for smooth surfaces and a large number of rough surfaces) the polarization Fresnel ratio can be used to determine the specular angle of incidence. Determination of the specular plane of incidence together with the specular angle of incidence uniquely specifies local surface orientation. We are currently proposing to build an integrated system of vision algorithms which simultaneously (i) classify material surfaces, (ii) separate reflection components, and, (iii) determine local surface normals; all from the determination of the polarization Fresnel ratio. The current name for the system is POLARIS which stands for POLarization And Radiometric Integrated System.

ACKNOWLEDGMENTS

The work done on classifying material surfaces using extended light sources was done in collaboration with Terry Boult.

REFERENCES

[Barrow and Tenenbaum 1978] Barrow, H.G., and Tenenbaum, J.M., *Recovering Intrinsic Scene Characteristics from Images*, in *Computer Vision Systems*, A.R. Hanson and E.M. Riseman, Eds., Academic Press, New York, 1978, pp.3-26.

[Cook and Torrance 1981] Cook, R.L., and Torrance, K.E., *A Reflectance Model For Computer Graphics*, SIGGRAPH 1981 Proceedings, Vol. 15 #3 pp.307-316, 1981.

[Healey and Binford 1988] Healey, G., Binford, T.O. *Predicting Material Classes, Proceedings of the DARPA Image Understanding Workshop*, April 1988.

[Healey and Blanz 1988] Healey, G., Blanz, W.E., *Identifying Metal Surfaces In Color Images*, SPIE 1988, April 1988, Orlando.

[Klinker et al. 1988] Klinker, G.J., Shafer, S.A., and Kanade T., *The Measurement of Highlights in Color Images*. International Journal of Computer Vision, Vol.2, No. 1, Spring, 1988.

[Physics Handbook] *American Institute of Physics Handbook*, Third Edition, McGraw-Hill, 1972.

[Shafer 1985] Shafer, S.A., *Using Color To Separate Reflection Components*, COLOR research and application, 10(4), pp.210-218, Winter 1985.

[Siegel and Howell 1981] Siegel, R. and Howell, J.R., *Thermal Radiation Heat Transfer*, McGraw-Hill, 1981.

[Torrance and Sparrow 1967] Torrance, K. and Sparrow, E., *Theory for Off-Specular Reflection from Roughened Surfaces. Journal of the Optical society of America*, 57, pp.1105-1114, 1967.

[Wolff 1987] Wolff, L.B., *Surface Orientation From Polarization Images*, SPIE Cambridge 1987, Optics, Volume 850, Illumination, and Image Sensing For Machine Vision II, pp.110-121.

[Wolff 1988a] Wolff, L.B., *Classification of Material Surfaces from the polarization of specular highlights*, SPIE Cambridge 1988, Optics, Illumination, and Image Sensing For Machine Vision III, pp. 206-213, 1988.

[Wolff 1988b] Wolff, L.B., *Segmentation of Specular Highlights From Object Surfaces*, SPIE Cambridge 1988, Optics, Illumination, and Image Sensing For Machine Vision III, pp. 198-205, 1988.

[Woodham 1978] Woodham, R.J., *Reflectance map techniques for analyzing surface defects in metal castings*, MIT AI Lab Tech Report AI-TR-457, June 1978.

Visual Homing Using an Associative Memory

Randal C. Nelson
Department of Computer Science
University of Rochester
Rochester, New York 14627

Abstract:

Homing is the process by which an autonomous system guides itself to a particular location on the basis of sensory input. In this paper, a method of visual homing using an associative memory based on a simple pattern classifier is described. Homing is accomplished without the use of an explicit world model by utilizing direct associations between learned visual patterns and system motor commands. The method is analyzed in terms of a pattern space and conditions obtained which allow the system performance to be predicted on the basis of statistical measurements on the environment. Results of experiments utilizing the method to guide a robot-mounted camera in a three-dimensional environment are presented.

Key words and phrases: visual navigation, 3-D vision, associative memory.

I. Introduction

Homing is the process by which an autonomous system guides itself to a particular location on the basis of sensory input. As stated, the definition can apply to motion in a variety of spaces; for instance, movement in the joint space of a robot arm. However, in this paper, we focus on controlling the motion of a single, compact sensor. In particular, we consider a compact observer equipped with an imaging sensor and able to move with ≤6 degrees of freedom in a complex (e.g., natural outdoor scene) three-dimensional environment which is assumed to be mostly rigid. A particular point in the motion space of the observer (position and orientation) is designated as the *goal point.* The visual homing problem is to utilize the information in the input image to direct the motion of the observer so that it can efficiently position itself in the neighborhood of the goal point from a starting point anywhere in a specified domain of competence.

Visual homing is an interesting problem for several reasons. From a theoretical standpoint, it is relatively well defined for arbitrary environments. This contrasts with navigational problems which are defined with respect to a specific environment (for example road following vehicles [Waxm87a]). Homing is thus potentially valuable as a general purpose component for navigation systems, in which it would operate in conjunction with lower-level systems providing (for example) stabilization and obstacle avoidance (see [Nels88a Nels88b, also, Ullm79, Praz81, Hild83, Waxm87b]). There are also a large number of potential practical uses for a visual homing system, including docking maneuvers, tool positioning, and grasp operations, as well as the obvious vehicle guidance applications. Finally, the problem is interesting because it appears to be one of the simplest visual operations performed by biological systems that involves substantial amounts of learned information. (For instance, certain bees and wasps are able to home visually in the immediate vicinity of their nests.) Because the mechanisms performing more complicated biological operations presumably developed from preexisting simpler ones, it seems plausible that a general mechanism for performing visual homing might provide a foundation for implementing and understanding more complex abilities.

There exist a number of automatic guidance systems which effectively implement homing, ranging from industrial robot controllers to missile guidance systems and automatic pilots. Many of these, however, use non-visual techniques. Robot controllers generally perform some sort of dead reckoning based on internally available coordinates; ICBMs do the same using an inertial guidance system, and a variety of automatic pilots simply triangulate on a set of radio beacons. Some surface-to-air missiles utilize an optical array to home on infrared signals emitted by the target, which could be considered a visual method. However, this generally amounts to tracking an unmistakable bright spot which does not require much sophistication in the way of image processing. There also exist several experimental systems which attempt to navigate on the basis of range images [Hebe88, Dail88], or by recognizing specific landmarks [Levi88]. In general, these methods involve maintaining an explicit map or three-dimensional model of the environment.

This paper describes a method of visual homing based on direct association between visual patterns and motor control. The basic idea is to store a large set of reference patterns, each of which represents a concise description of the environment as seen from the neighborhood of a particular position. Enough patterns are stored to cover the domain of competence. Associated with each reference pattern is information specifying an action to be taken (for instance, a direction of movement). Homing is accomplished by comparing an index pattern computed for the currently visible scene against the reference patterns and determining the best match. If the match is good enough, the scene is considered to be recognized and the associated action is executed. Note that such a system does not utilize an explicit model of the world; rather, the relevant characteristics of the environment are implicitly encoded in what might be viewed as the stimulus-response behavior of the system.

In order for this method to work, the transformation mapping images into the pattern space must possess several properties. First, the patterns produced for viewpoints close to each other in the motion space of the observer must be similar under an appropriate definition of similarity. This permits the reference patterns to span a neighborhood of non-zero volume, and allows the system to take advantage of the fact that similar motions will usually be required at similar positions. Second, in order to avoid spurious matches, the probability that highly similar patterns will be produced from unrelated viewpoints must be low. Third, the patterns must be relatively concise, as it is likely that many will have to be stored. The analysis of the principles involved in satisfying these design criteria is one of the major aims of this paper. Of particular interest is the possibility of obtaining quantitative estimates of the size of the recognition neighborhoods and the spurious match probabilities for various methods of pattern generation and different environments. These values can be used to determine whether or not the system will work under specific conditions.

This paper is organized as follows. Section II presents the basic definitions on which the system is based, Section III describes the design of the homing system, and Section IV presents an analysis which defines the conditions under which the system can be expected to work. These conditions are related to measurements which can be made on the environment and shown to hold for a selection of natural scenes. Section V describes the implementation and testing of the method using a robot-controlled camera in a complex three-dimensional environment, and Section VI presents conclusions and ideas for extending the work.

II. Basic Definitions and Operations

The system is based on two concepts: the notion of a pattern, and a definition of similarity between patterns which allows novel patterns to be compared to reference patterns in memory and to be recognized on the basis of a partial match. The definitions used are more or less classic, and were chosen to be as elementary as possible in order to facilitate analysis of the system. The idea is to show how these simple definitions can be used in conjunction with experimentally verifiable assumptions about the structure of the visual environment to implement a robust system of visual homing.

A *pattern* is defined to be a tuple $(a_1, a_2, \cdots a_m)$ which is an element of a *pattern space* $A_1 \times A_2 \times \cdots \times A_m$ where $A_1 \cdots A_m$ represent finite sets referred to as *features*. The idea behind this definition is the description of a scene in terms of discrete primitives. The features can be high level or low level, binary or multiply valued, depending on the desired characteristics of the pattern space. For example, a high level binary feature might be the presence or absence of a blob having a particular parameterization in the scene; a low-level multiply-valued feature might be the dominant edge direction in a particular receptive field.

The *similarity* $s(p_1, p_2)$ of two patterns which are elements of the same pattern space is defined to be the fraction of positions in the tuple at which the values are identical. Thus, for example, if the pattern space is chosen to be strings of length 3 over the English alphabet, then s(CAT, RAT) = 2/3 since the strings match in the last two positions, while s(ARM, MAR) = 0 since the strings match in no position, despite the fact that they contain the same characters. The similarity is related to the distance metric d_p, which is equal to the number of locations where the tuples differ, by the formula $s=(1-d_p)/m$ where m is the length of the tuples. The intuitive appeal of this definition is that it classifies visual objects as similar on the basis of the number of common features they contain.

The above definitions can be used to classify of patterns on the basis of a set of training samples as follows. First, store in memory all the sample patterns and their classifications. These constitute the set of *reference patterns*. When the system is presented with an unknown *index pattern*, it compares it against all the reference patterns, and extracts the one having the highest similarity. If this similarity is above a certain *recognition threshold* t, then the associated classification of the reference pattern is returned; otherwise, the system returns a "don't know" response. This can be considered to be a primitive associative memory, since classification is, in a sense, an associated pattern.

A memory constructed as above has an effective limit on the number of patterns that can be stored. Specifically, it is undesirable to store so many different patterns that an arbitrary "random" index pattern has a significant probability of yielding a response above the recognition threshold. This would lead not only to false recognition, but to mistakes in classification as well, since if a random pattern is likely to be recognized, it is also likely that the best match will be due to a chance coincidence and not the result of a meaningful similarity. The simplest way of avoiding such classification errors is to make the pattern space much larger than the domain of competence, and rely on statistical expectations to distribute the reference patterns across this space so that they do not interfere unexpectedly with each other. This idea is developed and made precise in the analysis section of this paper where it is used to derive measurements from which the performance of the system can be predicted.

III. Design Principles

The method of pattern classification described in the previous section can be adapted to the homing problem as follows. First, design an appropriate mapping Φ from images to a pattern space, and select an appropriate recognition threshold t. Then, at each of an appropriately chosen set **R** of reference points in the motion space, store in memory the corresponding pattern, and associate with that pattern either a direction of motion which will ultimately bring the observer closer to the goal, or information specifying that the goal has been reached. The set of patterns so stored constitute a set **P** of reference patterns, and the associated information can be viewed as the classification. Once the system has been trained by storing the set of reference patterns, homing is accomplished by applying Φ to the currently visible image to obtain an index pattern, and searching the memory to find the reference pattern that best matches. Assuming that a good match is found, the associated action is executed. This system operates without the use of an explicit global model of the world, utilizing only locally available information. Such a system is potentially extremely robust and, moreover, is more easily analyzable than one based on a global model, since all the interactions are local in nature.

The key to making the system work lies in the selection of the mapping, the recognition threshold, and the reference points. There are two basic conditions that must be met. First, the parameters must be selected so that every point in the space of competence C of the observer generates an index pattern which matches one of the reference patterns, i.e.

$$(\forall x \in \mathrm{C})(\exists y \in \mathbf{R}) s(P_x, P_y) \geq t, \tag{1}$$

where s is the similarity previously defined, and P_x and P_y are the patterns corresponding to points x and y respectively. For any point y in the reference set, the set of all local x such that $s(P_x,P_y) \geq t$ is termed the *recognition neighborhood* of y. Thus this condition simply states that the recognition neighborhoods of the set of reference points cover the domain of competence. The second condition is that the index pattern for a point x should match a reference pattern with antecedent point y only if x and y are near each other in the motion space. More formally, there must exist a function $r(x)$, sufficiently small for all points, such that

$$(\forall x)(\forall y) s(P_x, P_y) \geq t \rightarrow d(x,y) < r(x), \tag{2}$$

where d is a distance measure on the motion space. r can be thought of as representing the desired accuracy of the system. This condition prevents misclassification of points. The underlying assumption is that the variety of features present in complex scenes will, in general, be sufficient to allow a particular scene to be uniquely identified. In practice, what will be shown is that the probability of misclass ification is sufficiently small.

If these two conditions hold, and the appropriate actions are associated with the reference patterns, then the homing system is guaranteed to work in a static environment. If the probability of misclassification is nonzero but small, as will generally be the case in practice, then the system will work with high probability. In this case, the probability of error is bounded by the probability of making a misclassification on the path from the starting point to the goal. In fact, the situation is considerably better due to the fact that, since all reference is local, the system can generally recover from the effects of an occasional misclassification.

The two equations given above and the nature of the homing problem impose specific constraints which can be used to guide the design of the pattern space. First, the patterns produced for viewpoints which are close to each other in terms of movement of the vehicle should be similar, since similar motions will probably be required. This ensures that the reference points generate meaningful recognition neighborhoods. Note that the larger the recognition neighborhood, the fewer the number of patterns that must be stored to cover a given space and satisfy Equation 1. Second, and conversely, the patterns produced at widely separated locations should be dissimilar since dissimilar motions will probably be necessary. This is the essence of the constraint in Equation 2. Third, the pattern space must be large enough to hold the required number of viewpoints without impairing its "don't know" capacity. Fourth, the patterns should be relatively small in terms of the information required to describe them, since many will be stored. Thus, for example, storing the gray-level representation for the entire image would not be a good strategy. Finally, since the system is desired to work in real-world environments, the visual features chosen should be relatively insensitive to changes in lighting and to small perturbations in the physical environment.

A relatively simple idea which satisfies these constraints fairly well is to use the orientation of edges in local receptive fields. The basic strategy is as follows. First, identify edges in the image using some simple operator, and classify them into one of n (say 8) directions on the basis of the gradient direction at the edge points. Then divide the image into a set of receptive fields (for example, a 5×5 grid of squares), and determine the dominant edge direction in each field. The pattern consists of the vector representing the dominant orientation for each field. The scale of the edge detectors should be set so that, for well textured images, only a few lines run through each of the fields. The gradient estimators should be similarly scaled. This ensures that edges at the appropriate scale determine the pattern. Figure 1 illustrates the situation for a 5x5 grid of local receptive fields and a classification of the edges into 8 possible orientations. In this case, the patterns consist of vectors of 25 features, each of which can assume 8 values.

Such patterns are concise, and edges are relatively stable features under changes in lighting. The technical conditions are easily checked. Since the dominant orientation is determined by a few edges in each receptive field, the patterns will tend to change gradually with observer position as it will take a substantial change to move all of the edges out of their original fields. Thus the patterns behave as desired under observer motion. The size of the pattern space also seems to be adequate. For example, if the patterns are vectors of length 25 with 8 possible values, and the recognition threshold is 50%, then the probability that two patterns chosen randomly from a uniform distribution will have a similarity > t=50% is about 1.5×10^{-5}. For patterns of length 49 (corresponding to a 7×7 array of receptive fields) the value is about 5.5×10^{-10}. (These values can be computed from the cumulative binomial distribution.) Of course there is no guarantee that the patterns arising from real images will be uniformly distributed through this space – in fact, the existence of large-scale structure in the world would suggest otherwise; however, the scale of the probabilities suggests that there is enough room in the pattern space to deal with practical problems. The question of the actual distribution and the error probability in patterns taken from the real world is taken up in Section IV.

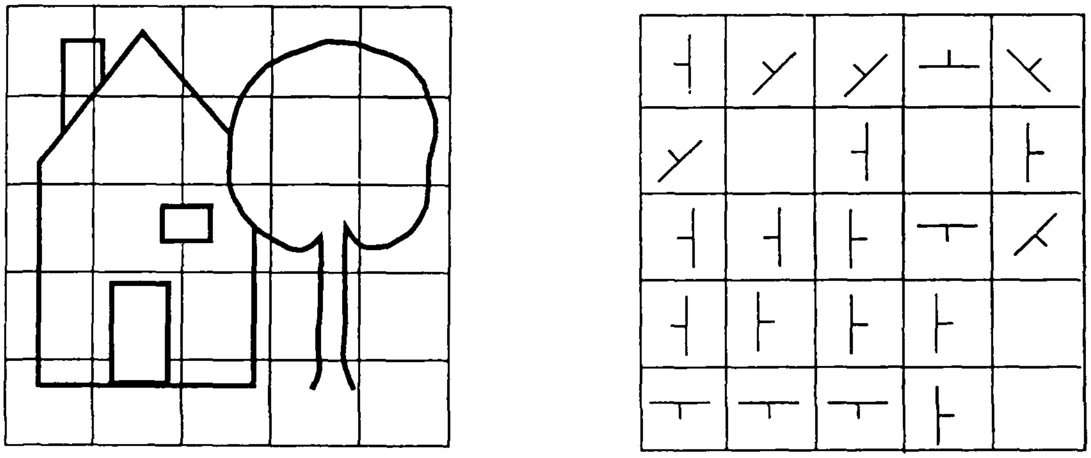


Figure 1: Edge image and pattern representing dominant edge direction in 5 × 5 array of receptive fields. The ticks in the pattern represent the direction of the gradient across the dominant edges.

Another issue that must be addressed is the question of the accuracy of the homing system. In general terms, the maximum accuracy with which the system can position itself with respect to any one of the degrees of freedom is approximately equal to the radius of the recognition neighborhood in that dimension. There is thus a basic trade-off involved in determining the size of the recognition neighborhoods. On the one hand, using a large neighborhood reduces the number of patterns that must be stored. On the other hand, the same large neighborhood reduces the accuracy of the homing system. One solution is to use a multiple resolution system. Far from the goal, a coarse resolution and large recognition neighborhoods can be used to approximately position the observer. Closer to the goal, information from a higher resolution set of features is used. By doing this at several levels, very high accuracy can be obtained over a relatively large domain of competence with the use of a modest amount of memory

Reducing the required amount of storage is an important issue for several reasons. In the first place, there may be physical limitations on the available memory which must be observed. In general, however, a more important reason for limiting the number of reference patterns is to minimize search effort, and to reduce the chance of exceeding the capacity of the pattern space. There are several strategies that can be used to improve the performance of the system in this respect.

A simple method of reducing the required memory is to utilize a physical search strategy to eliminate one or more degrees of freedom. For example, given a domain of competence encompassing one rotational and two translational degrees of freedom, one might store patterns only for a single value of the rotation parameter. If the system does not recognize the scene at the outset, it rotates until it does, and then moves in the remaining degrees of freedom. If the time needed to search the memory is short with respect to the time it takes to move the observer (as is the case with biological systems), then this could be an efficient strategy.

A more sophisticated method of reducing the search space and sparing the capacity of the pattern space is to make use of intermediate goals. The basic idea can be illustrated by considering the case of following a well-defined path. The path can be conceptually divided into a number of segments with an intermediate goal at the end of each. Along each segment, only the memory relevant to that portion of space is searched. When an intermediate goal is reached, the current memory is deactivated and the piece of memory describing the next segment is activated. Thus the information for one portion of the space does not interfere with information about another. The multi-resolution strategy described above is a special case of this.

The use of intermediate goals can be viewed as a simple method of utilizing the geometric structure inherent in the problem to organize the data. The idea can be generalized to link together separately described regions of space in more than one dimension. Carrying this process to its natural limit yields a "moving window" structure which organizes the memory at the granularity of the individual patterns over the full dimensionality of the motion space. This would be implemented by linking each pattern to the other patterns in the memory which correspond to points a short distance away in the motion space. During homing, these patterns are the only ones which would be accessed and compared. In a conventional machine, such an organization could be accomplished with pointers. In a neural implementation, excitatory links could be used to "preactivate" certain units. Use of the above method would virtually eliminate problems with saturation of the pattern space and with bounding the search. This is an important conclusion because it means that, in principle, the size of the domain of competence is limited only by the size of the physical memory and the efficiency of the encoding transformation.

IV. Analysis of the Associative Memory

In the previous section, two equations were presented which established conditions under which the homing method could be expected to work in a static environment. Informally stated, these conditions were first, that it be possible to store patterns corresponding to a set of reference points whose recognition neighborhoods cover the desired domain of competence, and second, that the probability of an index pattern being misclassified be sufficiently low. For a given pattern space and environment, there are two pieces of information which are critical to determining whether the conditions can be satisfied. The first is the size of the recognition neighborhoods. This will determine how many patterns must be stored to cover the desired domain of competence. The second is the probability p_f that two patterns corresponding to mutually distant points will match (i.e. have similarity $\geq t$). From these, the probability p_s of misclassification can be determined.

Recall that the recognition neighborhood of reference point is the set of all local points which produce patterns whose similarity to the reference pattern equals or exceeds the recognition threshold. For the purposes of visual homing, an attempt is made to design the pattern space so that the recognition neighborhood is a simply connected volume surrounding the reference point. In order to predict the number of reference points needed, and to determine their spacing, some estimate of the size and shape of the recognition neighborhoods must be obtained. In general, these quantities may depend upon the location of the observer in the motion space.

The size of the recognition neighborhood about a particular point in the motion space can be determined directly by moving the observer about and comparing the patterns at a large number of positions in the neighborhood of the point. By following such a procedure for many points in the motion space, an idea of the average sizes of the neighborhoods in various regions can be obtained. However, for spaces of more than two dimensions, this is a lot of work. A somewhat less arduous procedure would be to determine, either empirically or on the basis of knowledge about the features composing the patterns, the one-dimensional extent of the recognition neighborhood along each degree of freedom. The volume of the full neighborhood could then be estimated on the basis of these values. The naive formula would be simply to take the n-dimensional rectangle with principal axis lengths equal to the corresponding one-dimensional values. This corresponds to an n-ball in the motion space under a scaled max-length or chessboard metric. However, it ignores the cumulative effect of motion along different degrees of freedom, and produces an overly optimistic estimate of the volume. Use of a different metric can take this into account. It can be shown that, under reasonable assumptions, the recognition neighborhood will contain, in the worst case, an n-ball under an appropriately scaled sum or city-block metric, which corresponds to the n-dimensional analog of the octahedron. (Basically, one argues that the difference between patterns obtained at points x and y in the motion space is bounded by the change that can accumulate if the observer moves from x to y along a path that changes one coordinate at a time.) In practice, the neighborhoods tend to lie somewhere between the worst-case city-block n-ball and the Euclidean n-ball.

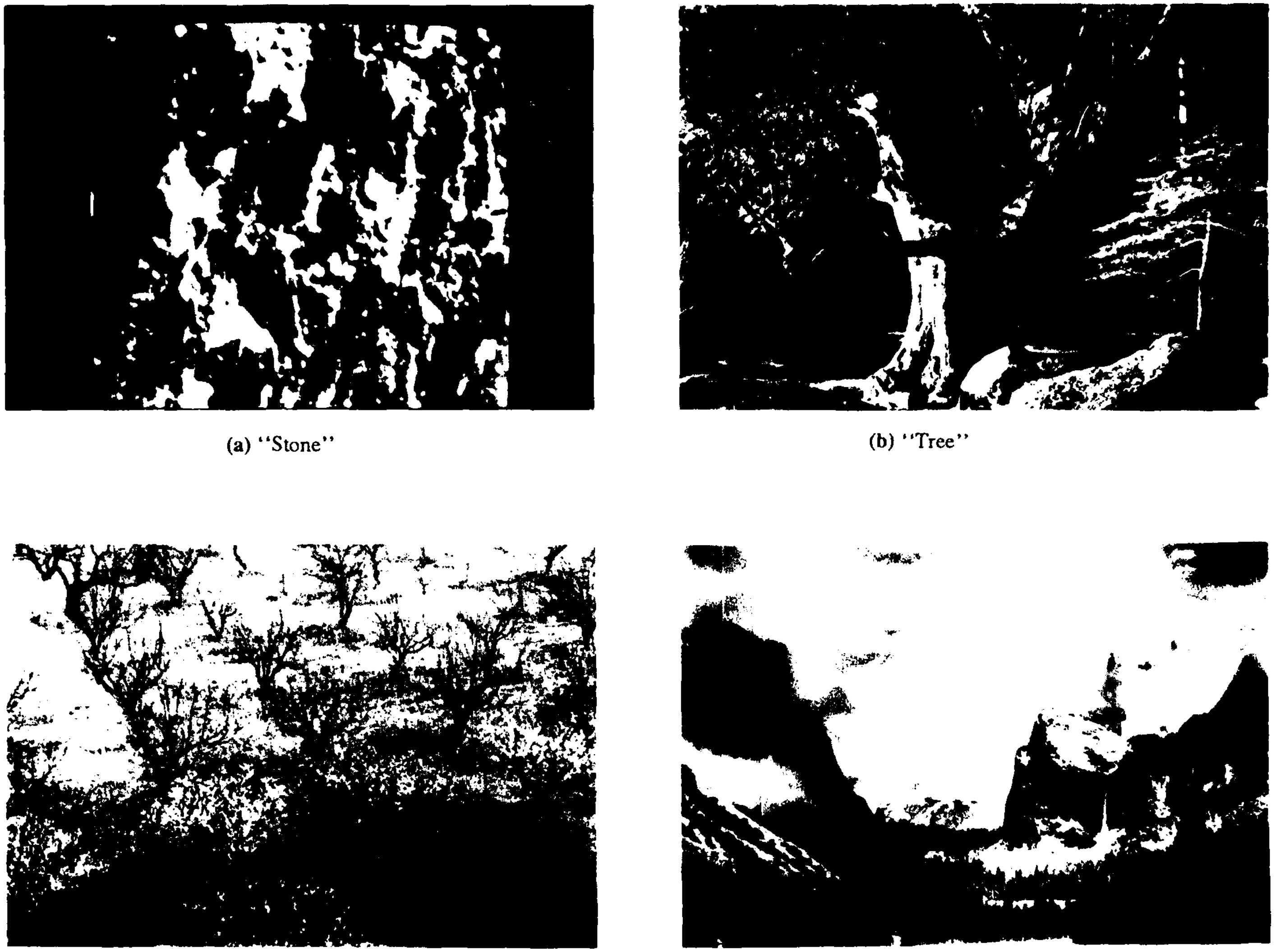
As an example, consider the problem of two-dimensional translation parallel to a flat, patterned surface. Let the pattern space be defined by edge orientation in an array of local receptive fields as described in the previous section. Since the edge detectors are designed to produce only a few lines crossing each field, the feature value is probably determined by one strong edge, and will change when that edge moves out of the field. The diameter of the recognition threshold can thus be estimated theoretically by computing the motion which would, on the average, cause a fraction $1-t$ of a randomly distributed set of points to move from their original fields. For translational motion, the decrease should be fairly linear; thus for a recognition threshold of .5, the recognition radius along each coordinate should correspond to a movement which causes the image to translate about halfway across the local receptive field.

The above situation can be easily simulated by windowing in a single large image. A series of tests were run using the four images shown in Figure 2. The pattern space utilized a 5×5 array of non-overlapping receptive fields each 9×9 pixels with a recognition threshold of .5. For each image, results for 100 different positions were combined to obtain an idea of the typical behavior in the neighborhood of a reference point. The arrays shown in Figure 3 depict, for each image, the average recognition neighborhoods in the two-dimensional motion space. Unit distance corresponds to a motion which would cause an apparent motion in the image of one pixel. The symbol "*" indicates the location of the reference point and "+" indicates locations where the scene was recognized. Note that the shape of the recognition neighborhoods fall between a diamond (city-block metric) and a circle (Euclidean metric). Also note that the radii, as predicted, are approximately the distance required to move the image halfway across the receptive field (here 4-5 units since a unit corresponds to a motion of one pixel).

In higher dimensions, some care must be taken in packing non-cubical regions together so that all the corners get filled. The severity of the problem becomes apparent when one notes that the proportion of the unit cube which is occupied by unit spheres under the other metrics decreases exponentially with increasing dimension (Table 1). For more information on n-dimensional geometry see [Coxe48] and [Somm29].

The second problem that must be addressed is showing that the probability of misclassifying a pattern is sufficiently low. Recall that the basic approach is to use a pattern space which is large enough to make it unlikely that an arbitrary pattern will be sufficiently similar to one of the reference patterns to trigger spurious recognition. The hope is to show that if an index pattern with antecedent point x matches a reference pattern with antecedent point y then, with high probability, x and y will be relatively close together in the motion space.

The first step is to define spurious similarity. As a shorthand, two patterns will be said to match if they have a similarity greater than the recognition threshold t. Now consider a scalar function $r(x)$ on the motion space which can be viewed as the desired accuracy of the system with respect to some distance metric d. If two patterns P_x and P_y with antecedent points x and y match, (i.e., $s(P_x,P_y)>t$), but $d(x,y)>\min(r(x),r(y))$, then the match is said to be spurious. In such a case, the similarity is presumably due to chance rather than physical proximity of the antecedent



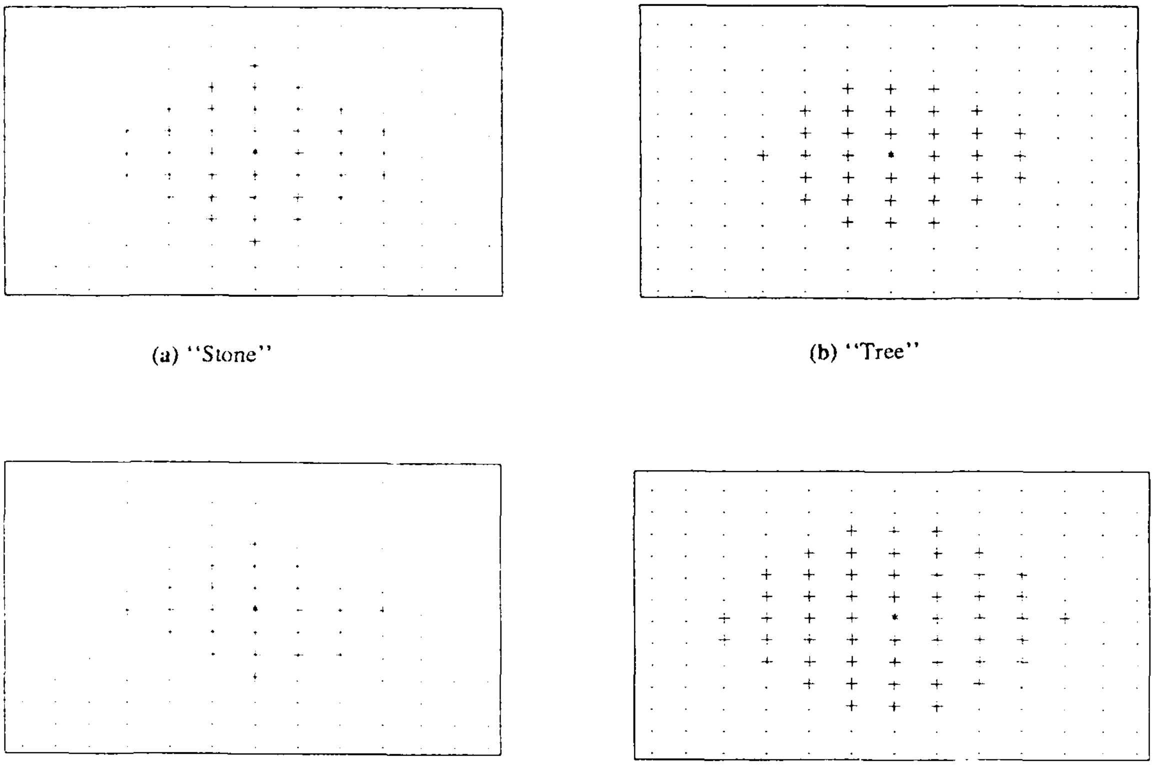
(a) ''Stone''

(b) ''Tree''

(c) ''Orchard''

(d) ''Storm''

Figure 2: Images used in statistical tests.



(a) "Stone"

(b) "Tree"

(c) "Orchard"

(d) "Storm"

Figure 3: Average recognition neighborhoods for images of Figure 2.

Table 1
Volume of *n*-balls of radius 1

Dimension	city-block	Euclidean	chessboard
1	2	2	2
2	2	π	4
3	4/3	$4\pi/3$	8
4	2/3	$\pi^2/2$	16
5	4/15	$8\pi^2/15$	32
6	4/45	$\pi^3/6$	64

points.

The function r provides the basis for a precise condition to be used in evaluating the probability of misclassification. Let x and y be an index point and a reference point respectively, and let P_x and P_y represent the corresponding patterns. Now let p_s represent the conditional probability that $d(x,y) \geq \min(r(x), r(y))$ given that $s(P_x, P_y) > t$, where d is an appropriate metric, and s is the pattern similarity. In other words p_s represents the probability that a given match is spurious. If it can be shown for a given r that

$$p_s < \mu \tag{3}$$

for some sufficiently small value μ, then a bound can be placed on the probability of the system making a navigational error due to misclassification of a pattern. Clearly r must contain the local recognition neighborhood if the condition is to be satisfied for reasonable values of μ. The best possible result would be to show that the relation holds for values of r which are just slightly greater than the radius of the recognition neighborhood.

One way to show that (3) holds is to compute the "spurious match" probability p_s directly for a given r, and show that it is sufficiently low. Two values are needed to do this: a "near match" probability p_n which is the probability that two patterns with antecedent points x and y will match given that $d(x,y)<\min(r(x),r(y))$, and a "far match" probability which is the probability that two patterns will match given that $d(x,y)\geq\min(r(x),r(y))$. In general, p_n is expected to be fairly large, and p_f very small. These values depend both on the structure of the pattern space and on the nature of the environment, and thus represent a real world component in the analysis.

Consider two domains A and B in the motion space. Later these domains will be interpreted as the sources of index and reference points. In general, their intersection will be non-zero, and in many cases they will represent the same region. Now consider a point pair x,y where $x\in A$ and $y\in B$. The probability p_m that the corresponding patterns P_x and P_y match is given by

$$p_m = p_f p_{d\geq r} + p_n p_{d<r} \tag{4}$$

where $p_{d\geq r}$ is the unconditional probability that $d(x,y)\geq\min(r(x),r(y))$ and $p_{d<r}$ is similarly defined. The probability p_s that the match is spurious given that a match exists is then

$$p_s = \frac{p_f p_{d\geq r}}{p_f p_{d\geq r}+p_n p_{d<r}} = \frac{p_f(1-p_{d<r})}{p_f(1-p_{d<r})+p_n p_{d<r}}. \tag{5}$$

Note that p_s is just the conditional probability $p_{d\geq r}$ given that P_x and P_y match. In general, the regions A and B will extend over distances much larger than r, so that $p_{d<r}$ is small compared to 1. Thus p_s can be approximated by

$$p_s \approx \frac{p_f}{p_f+p_n p_{d<r}}. \tag{6}$$

Furthermore, in the anticipated event that p_s is small, p_f must be small with respect to $p_n p_{d\leq r}$. Thus the further approximation

$$p_s \approx \frac{p_f}{p_n p_{d<r}} \tag{7}$$

can be made.

The value of $p_{d<r}$ must now be determined. It was noted above that, in general, the regions A and B will extend over distances considerably larger than r. This will also be the case for the intersection $A\cap B$. If this is true then, for reasonably configured regions (i.e, no pathologically shaped common boundary), the distance between x and y will, in most cases, be less than r only if both x and y lie in the region of intersection. If the distribution of x in A and y in B is uniform, and the two points are assumed to be independent, then the probability that x lies in the region of intersection is given by $|A\cap B|/|A|$ where the magnitude corresponds to volume, and likewise for y. If the distribution of points within the regions is not uniform, the same formula can be used with volume replaced by a weighted integral. Within the region of intersection, the probability $p_{d<r}$ is the ratio of the average volume v_r of a ball of radius r in that region to the total volume of the intersection region itself. Hence

$$p_{d<r} \approx \frac{|A\cap B|^2 v_r}{|A||B||A\cap B|} = \frac{|A\cap B| v_r}{|A||B|}. \tag{8}$$

If A and B represent the same region, this reduces to v_r/v_{tot} where v_{tot} is the volume of the entire region of interest.

Combining (7) with (8) yields the equation

$$p_s \approx \frac{p_f |A||B|}{p_n |A\cap B| v_r}. \tag{9}$$

Since the recognition neighborhood will, in general, be contained by the r-neighborhood, p_n is given by the average volume ratio of the two neighborhoods. In general, r can be selected so that this ratio is approximately constant, in which case $p_n \approx v_{rn}/v_r$ where v_{rn} is the average volume of the recognition neighborhood. Hence

$$p_s \approx \frac{p_f |A||B|}{v_{rn} |A \cap B|}, \tag{10}$$

and in the case that the two regions are the same

$$p_s \approx \frac{p_f v_{tot}}{v_{rn}}. \tag{11}$$

Thus the probability of misclassification by the system is approximately the minimum number of recognition neighborhoods required to cover the region of interest, times the probability of two patterns matching accidently. This is more or less what would be expected intuitively.

Now let A correspond to the source domain from which index points are to be drawn, and let B correspond to the domain of competence covered by the recognition neighborhoods of the reference points. In practical applications, these regions will be well behaved in the respects required by the preceding derivation, which means that p_s can be calculated using (10) or (11) and used to show that (3) is satisfied.

It remains to determine a value for the "far probability" p_f. Recall that p_f represents the probability that two patterns whose antecedent points are separated by a distance greater than r will match. Intuitively, it corresponds to the match probability for "unrelated" patterns, that is, patterns for which any similarity is due to chance rather proximity of the antecedent points in the motion space. The simplest approach is to assume first, that patterns for points separated by $d > r$ are uncorrelated, and second, that the patterns obtained from the environment are uniformly distributed over the pattern space. In this case, p_f can be computed theoretically from the characteristics of the pattern space. The first assumption is reasonable, since a pattern space which works well for the homing application will tend to have this property. For example, in the edge-based pattern space discussed in Section III, the long-range correlation of edges, at least in messy natural images, is fairly low. The second assumption is questionable, since it implies that there is also no short-range correlation of the features. It is certainly suspect for the edge-based space of Section III, since short-range correlation of edges will be found in almost any image. The value for p_f computed under the assumption of uniform distribution represents a best case, i.e., p_f can not possibly be any smaller. This is a consequence of the information-theoretic result which states that the ensemble having maximum entropy is a uniform distribution (see for example [Gall68]). It thus has value as a screening test for a proposed system. If the p_f is insufficiently small under the above assumptions, then the system certainly will not work.

For the pattern spaces of Section III, the computation of the match probability for patterns drawn independently from a uniform distribution is straightforward. The patterns consist of n independent features each of which can take on m values with equal probability. They can thus be thought of as strings of length n over an alphabet of m characters. The probability that two patterns selected independently from such an ensemble will match in k or more positions is just the value of the cumulative binomial

$$\sum_{i=k}^{n} \binom{n}{i} (1/m)^i (1-1/m)^{n-i}. \tag{12}$$

Table 2 lists the value of this expression with m=8 for a number of pattern lengths (n) and several similarity coefficients (k/n). As can be seen from the table, there is a great deal of space available for modest sized patterns and reasonable (e.g. 50%) recognition thresholds. Moreover, the match probability at a given recognition threshold decreases exponentially with increasing pattern size. Hence as long as additional features with some degree of independence can be added, there should be little trouble devising a pattern space of sufficient size for any particular problem, assuming that the features used are adequate in other respects.

As was mentioned above, the ensemble of patterns generated from the environment is unlikely to be uniformly distributed, and consequently the value of p_f in a real application tends to be higher than would be predicted from the above formula. The actual departure from the uniform value varies considerably depending on the particular environment. Since it seems unlikely that an abstract model for natural scenes can be devised which does not at least use empirically determined parameters, obtaining an approximation for p_f will probably require empirical measurements of some sort. The most direct method of determining p_f is simply to estimate the match probability statistically by comparing the patterns corresponding to a (large) number of points separated by distances greater than r. The required number is not actually as large as might first be supposed. k points will produce on the order of $k^2/2$ interactions, so probabilities on the order of 10^{-5} could be reliably estimated, and upper bounds on the order of 10^{-6} obtained on the basis of a few thousand patterns.

Table 2
Probability of 100*k*/*n* percent match between two random strings of length *n* over alphabet of size 8.

	Length									
100*k*/*n*	10	20	30	40	50	60	70	80	90	100
10%	7.4×10^{-1}	7.3×10^{-1}	7.4×10^{-1}	7.5×10^{-1}	7.6×10^{-1}	7.8×10^{-1}	7.9×10^{-1}	8.0×10^{-1}	8.1×10^{-1}	8.2×10^{-1}
20%	3.6×10^{-1}	2.3×10^{-1}	1.6×10^{-1}	1.2×10^{-1}	8.8×10^{-2}	6.6×10^{-2}	5.0×10^{-2}	3.8×10^{-2}	2.9×10^{-2}	2.2×10^{-2}
30%	1.2×10^{-1}	3.1×10^{-2}	9.0×10^{-3}	2.7×10^{-3}	8.6×10^{-4}	2.7×10^{-4}	8.7×10^{-5}	2.8×10^{-5}	9.2×10^{-6}	3.0×10^{-6}
40%	2.7×10^{-2}	1.8×10^{-3}	1.4×10^{-4}	1.1×10^{-5}	9.3×10^{-7}	7.8×10^{-8}	6.7×10^{-9}	5.7×10^{-10}	5.0×10^{-11}	4.3×10^{-12}
50%	4.4×10^{-3}	5.2×10^{-5}	6.8×10^{-7}	9.6×10^{-9}	1.4×10^{-10}	2.0×10^{-12}	3.0×10^{-14}	4.5×10^{-16}	6.8×10^{-18}	1.0×10^{-19}
60%	5.1×10^{-4}	6.9×10^{-7}	1.1×10^{-9}	1.7×10^{-12}	2.9×10^{-15}	5.0×10^{-18}	8.6×10^{-21}	1.5×10^{-23}	2.7×10^{-26}	4.7×10^{-29}
70%	4.0×10^{-5}	4.2×10^{-9}	4.9×10^{-13}	6.2×10^{-17}	8.0×10^{-21}	1.0×10^{-24}	1.4×10^{-28}	1.9×10^{-32}	2.5×10^{-36}	3.5×10^{-40}
80%	2.1×10^{-6}	1.0×10^{-11}	5.8×10^{-17}	3.4×10^{-22}	2.1×10^{-27}	$1.3\times10^{\ \ 2}$	8.2×10^{-38}	5.2×10^{-43}	3.4×10^{-48}	2.2×10^{-53}
90%	6.6×10^{-8}	$8.2\times10^{\cdot15}$	1.1×10^{-21}	1.7×10^{-28}	2.5×10^{-35}	3.9×10^{-42}	6.1×10^{-49}	9.6×10^{-56}	1.5×10^{-62}	2.4×10^{-69}

A number of experiments were performed to compare the values of the "false match" probability p_f for pattern ensembles based on real images to the values computed for a uniform distribution. Three edge-based pattern spaces were used, utilizing 5×5, 6×6 and 7×7 arrays of local receptive fields. The same two-dimensional scenario that was used to test the local similarity was used here, the only difference being that the sample points were separated by sufficient distance so that no point in the image contributed to the same feature in two different patterns. Measurements were made for the four images used previously. Regions without visible features, such as uniform sky, were excluded from the analysis. Approximately 6000 patterns were used in computing p_f for the tree, orchard, and storm images, and approximately 2500 for the stone image. 6000 patterns generate on the order of 2×10^7 interactions; however, the conciseness of the patterns allowed all the comparisons to be made in a reasonable amount of time. Table 3 summarizes the results of the experiments.

As expected, the match probabilities are higher than would be predicted from a uniform distribution, with the magnitude of the difference varying according to the image used. The probabilities for the orchard image are smallest, and thus closest to the uniform values for both pattern spaces, probably because the picture is full of very fine detail with little correlation at the scale of the operators used. In general, the match probabilities are low enough to allow a considerable number of patterns to be stored with little risk of misclassification. The table also illustrates the dramatic effect of a modest increase in the size of the pattern. Going from a 5×5 array to a 6×6 increases the

Table 3
Match probability for "unrelated" patterns

Source	5×5	6×6	7×7
Uniform	1.5×10^{-5}	3.2×10^{-8}	5.5×10^{-10}
Stone	6.5×10^{-5}	3.6×10^{-6}	$<10^{-6}$
Tree	5.2×10^{-5}	6.3×10^{-6}	3.3×10^{-6}
Orchard	2.8×10^{-5}	1.7×10^{-7}	$<10^{-7}$
Storm	1.1×10^{-4}	4.0×10–6	9.1×10^{-7}

storage requirements by less than 50%, but results in a 10-fold or greater increase in the number of patterns that can be stored at a given misclassification probability. This suggests that, at least for edge-based pattern spaces of the type considered here, reducing the chance of spurious classification to an acceptable level is not a major problem.

Using the above results, it is possible to predict the size of the domain of competence that can be covered. Table 3 indicates that an edge-based pattern space utilizing a 7×7 array of receptive fields (49 features) would permit on the order of 10^4 reference patterns to be stored with only a small (1%) chance of misclassification. This is also about the maximum number that could be comfortably handled by our experimental system. The question is, what kind of domain can be covered using ten thousand reference points? The answer depends on the nature of the desired domain of competence.

As a first example, suppose that the motion space corresponds to two-dimensional translation at constant altitude with a downward looking observer over an environment where the vertical relief is relatively small compared to the altitude of the system (Figure 4). In order to make the results meaningful, suppose that the motion space is scaled so that one unit corresponds to a movement which will cause an image shift the diameter of the visual field. One unit of area in the motion space thus corresponds to a "camerafull". If the image contains 7×7 visual fields, the recognition neighborhood would be expected to have a diameter of about 1/7 unit, and an area (if the worst-case city-block metric is assumed) of about $1/2 \times 1/7 \times 1/7 \approx 1/100$ square units. 10^4 reference points will thus permit a domain of competence corresponding to about 100 camerasfull. This is fairly sizable in a local sense. In concrete terms, it corresponds to learning about 4 square miles from an altitude of 1000 feet with a 60°×60° field of view.

As a second example, consider the same situation but with a rotational degree of freedom added to the motion space. A rough calculation of the rotation that would cause the principal edges to move out of half of the receptive fields yields a ballpark figure of 1/25 of a revolution (15°) for the diameter of the recognition neighborhood along the rotational degree of freedom. The volume of the recognition neighborhood, (again assuming the worst-case city-block metric) is $1/6 \times 1/25 \times 1/7 \times 1/7 \approx 1/7500$ revolution-camerasfull. 10^4 reference points would just cover one camerafull over a full rotational degree of freedom. This would be sufficient for performing local docking maneuvers, but the result also indicates that the edge-based patterns are not particularly appropriate for performing coarse orientation movements in several degrees of freedom. A better plan would be to use a pattern space having features with considerable invariance under rotation to roughly orient the observer, and then use an edge-based system to perform accurate maneuvers. Such coarse pattern spaces might utilize higher level primitives, for instance, parameterized blobs or colored regions.

As a third example, consider the problem of navigating a course along a system of roads. The homing system must keep the vehicle centered in the road, and negotiate turns at the proper time. The domain of competence can thus be considered as a "fat" one-dimensional space. The primary extension is along the forward translational degree of freedom, but a small buffer zone in the other two degrees of freedom (one rotational, one translational) is needed to direct the vehicle back to the correct path should it start to wander off. A 3×3 extension in these two dimensions should be sufficient, and resultant domain can be viewed as a "worm" of diameter 3 wandering through an (x, y, ω) motion space (Figure 5). If the camera has a 45° field of view, and it is assumed that there is scenery along the sides of the road, then rough calculations indicate that the diameter of the recognition neighborhood in the direction of forward motion for a 7×7 array of edge features is approximately $w/7$ where w is the width of the road. In this case, most of the diameter can be used, which leads to a domain of competence of length $10^4 \times 1/9 \times 1/7 \times w \approx 100w$. For a road width of 50 feet, this is about a mile. If a special purpose system were available for keeping the vehicle on the road, and the memory were used only to recognize regions within (say) 100 feet of where a turn had to be made, then a longer course could be navigated.

V. Implementation and Testing

A system based on the principles described in Section III was implemented, and its performance tested in a visually complex domain. A mobile observer was implemented by attaching a small ccd camera to a robot arm. By appropriate motion of the robot, the camera can be moved about the environment with six degrees of freedom.

The pattern association system utilizes an edge-based pattern space of the type described in Section III. Specifically, the patterns are composed of 25 features which correspond to the dominant edge direction in a 5×5 array of non-overlapping receptive fields. Generally, the feature value indicates one of eight possible directions, but there is also a ninth "don't know" value which is used to indicate insufficient information. The recognition threshold was set at 12 matches, i.e., approximately .5. The array is scaled to cover the 35°×48° field of view, and thus is useful for homing at rather coarse resolution. If better ultimate resolution is required, a smaller subfield could be

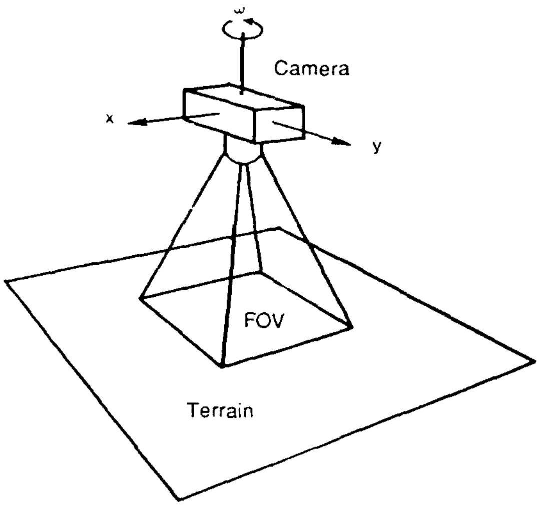


Figure 4: Geometry for observer moving over landscape in three degrees of freedom.

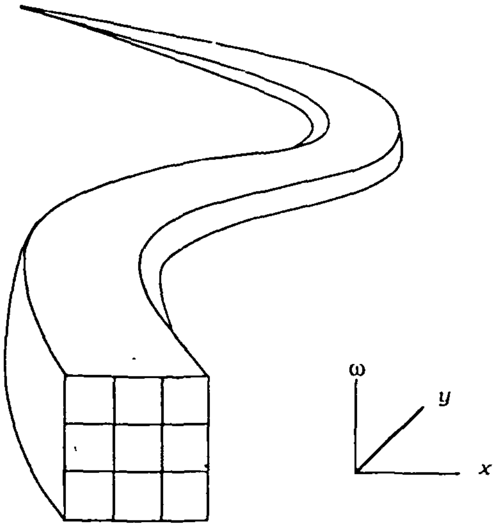


Figure 5: Restricted domain of competance in three degrees of freedom sufficient to guide an observer along a particular path such as a road.

used for fine-tuning the position near the goal. The results of Section IV indicate that this pattern space could be expected to have a p_f somewhere in the range from 10^{-5} to 10^{-4}, which would permit on the order of 100 patterns to be stored with less than a 1% chance of spurious identification. This capacity was sufficient to permit the initial experiments to be successfully carried out.

The system memory is created by a *training procedure*. In this phase, the camera is positioned at a number of reference points in the motion space. At each point, a picture is taken by the camera, and the corresponding pattern extracted by the image processing software. Also at each point, the system determines what action is appropriate to take at that location (e.g., "stop, the goal has been reached", or "move one inch in the *z* direction"). The pattern and a code for the associated action are then stored as a "memory trace".

Once the system memory has been established by sufficient training, the following procedure is used for homing. First, an image is acquired from the camera and the corresponding index pattern computed. This pattern is then compared against the reference patterns in memory, and the one having highest similarity is extracted. If the similarity is higher than the recognition threshold, the associated action code is retrieved. In this implementation, the code either specifies a direction of motion or indicates that the goal has been reached. If movement is indicated then the robot is instructed to move the observer a small distance in the indicated direction. This incremental motion minimizes the consequences should spurious recognition of a pattern occur. Several heuristics are included to make the system more robust. For instance, if no match is found for the index pattern, an additional move is made in the previous direction in the hope of skipping over the "blind spot . If no previous information is available, the system enters a random search. Similarly, two successive goal-reached codes are required for the system to conclude it has indeed reached its goal.

The procedure described above is actually a discrete simulation of a much finer-grained "continuous" control procedure which could be used if the cycle time were fast enough to permit real-time dynamic operation. In this case, the action code would specify a desired velocity (direction and magnitude) rather than simply a direction of motion. The motion persistence, which was included rather artificially in the discrete procedure in order to make it more robust, would tend to arise naturally from the dynamics of the system in the continuous case.

As a test of the system, a model environment fancifully christened "Tinytown" was constructed using HO scale scenery: trees, bushes buildings, etc. The model, is shown in Figure 6. The robot was configured so that the camera pointed towards the "ground" and was constrained to move in a two-dimensional motion space parallel to the this ground plane. The resulting images thus simulate low-altitude aerial photographs. Figure 7 shows one such image and the results of various stages of processing. Figure 7a shows the original image, 7b the reduced resolution version, 7c the results of edge extraction, and 7d the 5×5 pattern of dominant edge directions.

The model environment consists of about four camerasful at the selected elevation. In order to exercise the ability of the system to deal with occasional unclassifiable patterns, a slightly coarse sampling grid was chosen for training. Approximately 120 reference points were needed to cover the two-dimensional motion space. A goal position was selected in which the camera was approximately centered over the model and the patterns corresponding to the 3×3 block of reference points surrounding this positions were assigned goal-reached action codes. The rest of the reference patterns were assigned one of eight directional codes corresponding to the straight-line direction to the goal.

Figure 8 shows the results of testing the system on a grid with twice the sampling rate of the reference grid in a region surrounding the goal. The figure indicates the value of the action code determined for each point, or whether the pattern was unclassifiable. As expected, because of the coarse spacing of the reference points, a fair number of the points (about 25%) yield unclassifiable patterns. However, of the points where an action code was returned, only one has a value which would misdirect the system. This is consistent with the estimate mentioned at the beginning of this section, that the pattern space could result in a 1% spurious classification rate for 100 stored patterns.

The information provided by the memory, even with the coarse reference sample, is sufficient to permit the system to home reliably using the procedure discussed previously as can be seen from Figure 8. Figure 9 shows a typical path taken by the system in returning from the periphery of the domain of competence to the goal. In this case, the step size was set to .7 times the sample spacing to prevent a misleading positive result. Note that the goal zone represents a fairly extensive region. If a more accurate final position is required, the above procedure would represent only the first step of a multiresolution process. Instead of halting upon reaching the goal zone, the system would perform a context switch, activating a different associative memory whose domain of competence approximately encompasses the goal zone, and a higher resolution set of sensors.

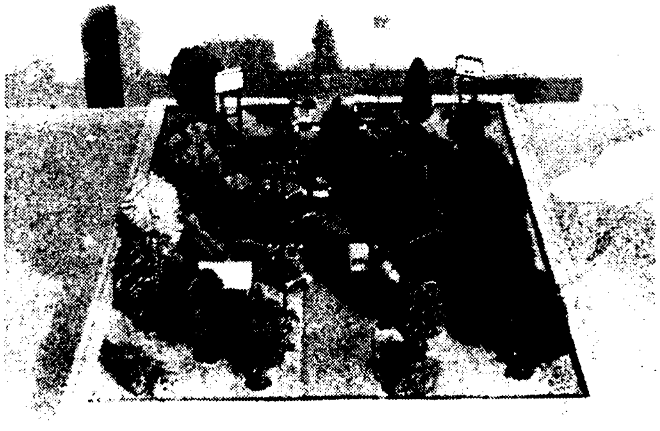
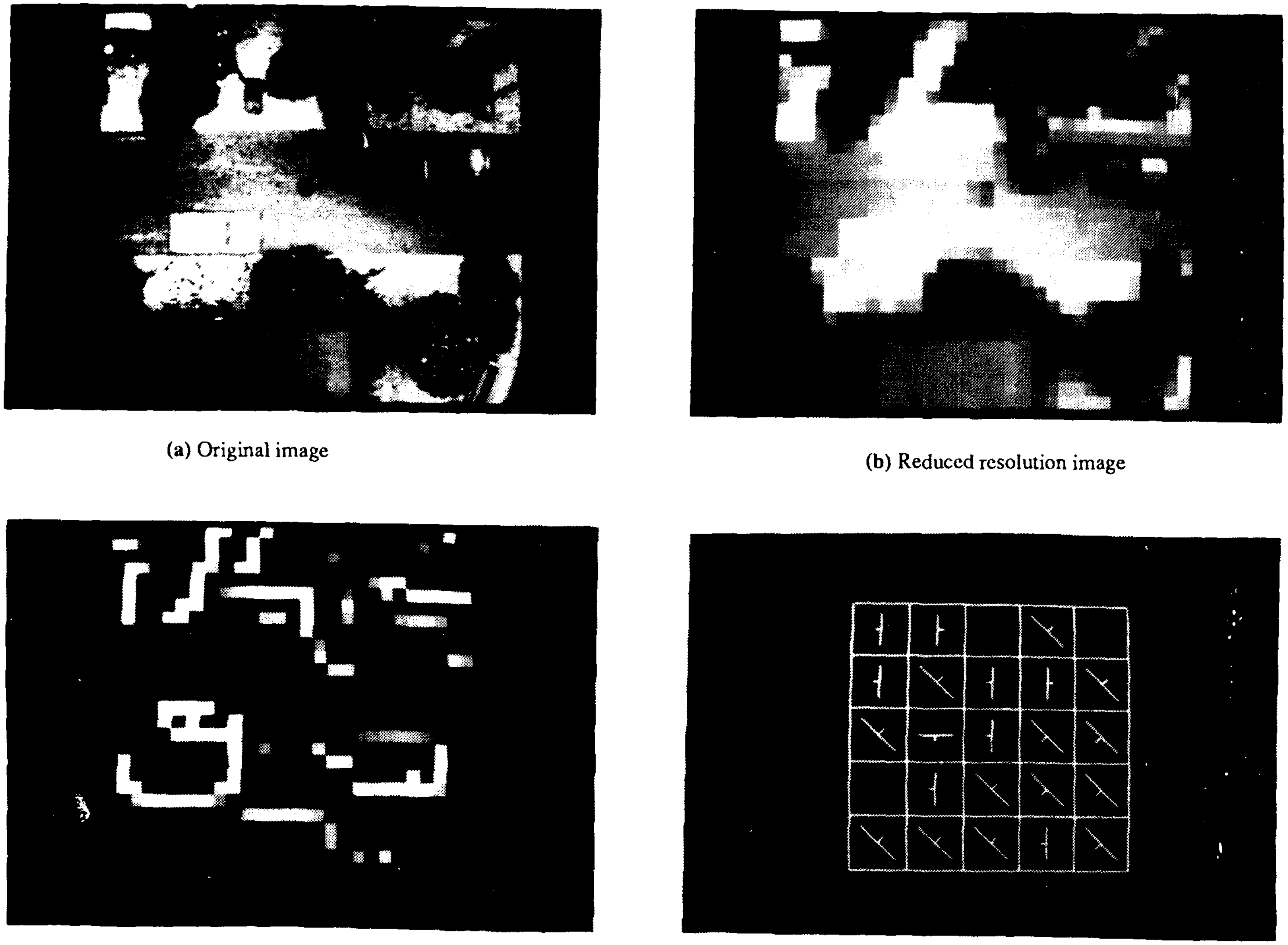


Figure 6: "Tinytown"



(a) Original image

(b) Reduced resolution image

(c) Edge image

(d) Pattern

Figure 7: Processing steps in reducing image to pattern.

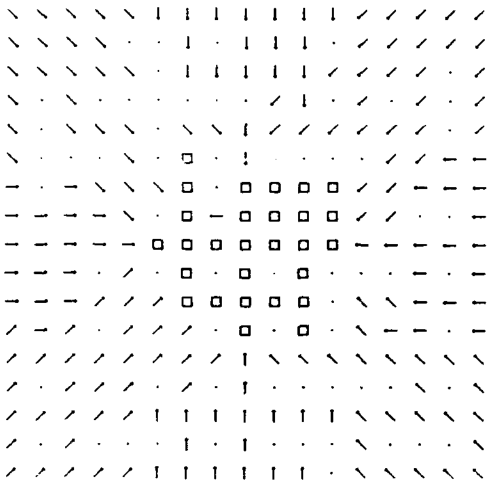


Figure 8: Results of testing associative memory for homing in tinytown. The sampling interval is 1/2 inch, which subsamples the one-inch grid of ref^rence p‹ ..its. Arrows indicate direction of associated motion, squares a goal-reached condition, and dots a "don't know" condition.

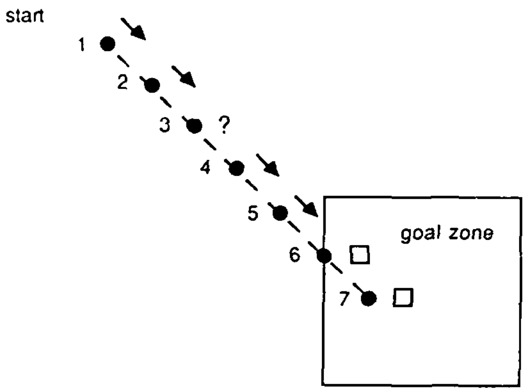


Figure 9: Typical path followed by robot. Numbers indicate indicate the sequence of positions. The symbols indicate the action code recovered from the associative memory at each position. Arrows indicate a direction of movement, boxes a goal-reached condition, and question marks a "don't know" condition.

The above results support the position that an associative memory based on the principles presented can be used to implement visual homing in a general, complex environment. The results of the analysis and empirical experiments involving the storage of thousands of reference patterns carried out in Section IV are persuasive evidence that the system could be made to work in a more extensive environment. The only change necessary would be to use a slightly larger pattern, say one based on a 7×7 array of receptive fields, in order to provide more capacity in the pattern space.

VI. Summary and Directi^ns for Future Work

This paper has argued that visual homing constitutes a basic navigational operation for which a general solution exists. In particular, a method of visual homing based on the direct association of visual patterns with observer motions has been described. The basic idea is to store a large number of reference patterns, each of which is a concise representation of what the world looks like in the vicinity of a particular reference point, and to associate with each pattern a movement which will bring an observer at the antecedent point closer to the goal. By storing a sufficient number of reference patterns, it is possible to associate an arbitrary index point (within a specified domain of competence) with nearby reference points on the basis of pattern similarity, and thus obtain an appropriate motion. The proposed method relies on the fact that there exist mappings which take images into a concise pattern space in such a way that the patterns corresponding to images from close pairs of points are similar, and the patterns corresponding to images from distant pairs are unlikely to be similar.

The paper provided an analysis which identified explicit conditions under which the method could be expected to work. It was shown that both the required number of reference points and the probability of error can estimated on the basis of the structure of the pattern space and measurable properties of the environment. General high-level considerations suggested that a group of edge-based pattern spaces might represent one class of mappings having the required properties. Large scale statistical tests run on a variety of natural images tended to support this position. On the basis of the tests and analyses, it was concluded that a homing system using an edge-based pattern space would be practical for performing accurate navigation in "fat" one- or two- dimensional spaces. Problems such as city street navigation can be put in this form. Coarse orientation in several dimensions seems to require a different pattern space. As a practical demonstration, the method was implemented using a camera mounted on a robot arm to navigate in a model environment. The system was able to home successfully in a complicated 3-D environment.

The proposed method has several advantages. First, it is a general navigational technique. It can be applied in any environment which has sufficient visual variety in a sense which has been well defined. Training for a particular application is accomplished using a set of examples which are selected on the basis of a well specified set of principles. Second, the method is quite robust, both in terms of dealing with small changes in the environment, and in terms of recovering from the results of misclassification or missing information. This robustness is due both to the qualitative nature of the pattern matching process, and to the redundant formulation of the problem which results in similar information being available at neighboring points. Thus if a mistake is made at some point, or the necessary information is unavailable, it is likely that correct information will be provided nearby. Finally, the method is easily realizable in a massively parallel architecture, which makes it a candidate for real-time implementation.

This project sparked a large number of ideas for continuing research. One avenue concerns the identification of pattern spaces which are appropriate for handling large-scale rotational orientation at a coarse resolution. This leads directly to applications in 3-D object recognition. Another avenue concerns the apparent generalizing ability of systems utilizing the sort of associative memory described here. In one experiment, the system was trained to home on the noses of 16 different faces. It was then able to successfully home onto the nose of a face it had not seen before. This kind of behavior has implications for automatic generation of special purpose systems such as road followers. A third avenue involves the consequences of attempting to implement the method using a neural paradigm. In order to do this in a reasonable manner, it turns out that modifications must be made to the memory storage procedure which have the side effect of causing the system to organize the information in a way that can be interpreted as automatic generation of classes. All three avenues appear potentially fruitful.

References.

[Coxe48] - H. S. M. Coxeter, *Regular Polytopes*, Methuen & Co. Ltd., London, 1948.

[Dail88] - M. Daily, J.G. Harris, and K. Reiser, An operational perception system for cross-country navigation, Proc. DARPA Image Understanding Workshop, April 1988, 568-575.

[Gall68] - R. G. Gallager, *Information Theory and Reliable Communication*, John Wiley and Sons, New York, 1968.

[Hebe88] - M. Hebert and T. Kanade, 3-D Vision for outdoor navigation by an autonomous vehicle, Proc. DARPA Image Understanding Workshop, April 1988, 593-601.

[Hild83] - E. C. Hildreth, *The Measurement of Visual Motion*, The MIT Press, Cambridge Mass, 1983.

[Levi88] - S. Levitt, D. T. Lawton, D. M. Chelberg, K. V. Koitzsch, and J. W. Dye, Qualitative navigation II, Proc. DARPA Image Understanding Workshop, April 1988, 319-326.

[Nels88a] - R.C. Nelson and J. Aloimonos, Finding Motion Parameters from Spherical Flow Fields (Or the Advantages of Having Eyes in the Back of Your Head) *Biological Cybernetics* **58** 1988, 261-273.

[Nels88b] - R.C Nelson and J. Aloimonos, Using flow field divergence for obstacle avoidance in visual navigation, Proc. DARPA Image Understanding Workshop, April 1988, 548-567.

[Praz81] - K. Prazdny, Determining the instantaneous direction of motion from optical flow generated by a curvilinear moving observer, *CVGIP* **22**, 1981, 238-248.

[Somm29] - D. M. Y. Sommerville, *An Introduction to the Geometry of N Dimensions*, Methuen & Co. Ltd., London, 1929.

[Ullm79] - S. Ullman, *The Interpretation of Visual Motion*, The MIT Press, Cambridge, Mass, 1979.

[Waxm87a] - A. Waxman, J. LeMoigne, L. Davis, E. Liang, and T. Siddalingaiah, A visual navigation system for autonomous land vehicles, *IEEE Journal of Robotics and Automation*, 1987.

[Waxm87b] - A. Waxman, Image Flow Theory: A framework for 3-D Inference from Time Varying Imagery, in *Advances in Computer Vision*, C. Brown editor, Lawrence Erlbaum Associates Inc, 1987.

Representation Space: An Approach to the Integration of Visual Information

Aaron F. Bobick Robert C. Bolles

Artificial Intelligence Center
SRI International
333 Ravenswood Ave.
Menlo Park, CA 94025

Abstract: An approach to representing objects viewed over long periods of time and with changing resolution is presented. The basic strategy is to apply different representations as they become appropriate. As a result, the model of an object typically goes through a sequence of representations as new data are gathered and processed. One of these sequences might start with a crude blob description of an initially detected object, include a detailed structural model derived from a set of high-resolution images, and end with a semantic label based on the object's description and the sensor system's task. This evolution in representations is guided by a structure we refer to as *representation space*: a lattice of representations that is traversed as new information about an object becomes available. One of these representations is associated with an object only after it has been judged to be valid. We describe one approach for evaluating the validity of an object's description that is based on the temporal stability of the description. We illustrate these ideas with results from the TraX system which constructs and refines models of outdoor objects detected in sequences of range data.

The Problem: Temporal Integration of Information

The problem we address is how to incrementally construct a description of an object as new information about that object becomes available. Our approach combines well-known quantitative strategies [1,4,7] with a more qualitative technique that permits the underlying representation to change over time.

To illustrate why the temporal integration of information is important, consider the example of an autonomous robot navigating off-road terrain. One approach to this task is to have the robot analyze one range image at a time, completely independently [2]. At each time step, the vehicle captures a range image, computes the scene geometry, plans a short, clear path toward the goal location, and then starts down that path. A potential flaw with this approach is that a single mistake in processing an image can produce disastrous results. For example, assume that the robot's algorithms occasionally make a mistake of omission — not detecting an obstacle in the field of view. If the robot plans the next path based solely on the current image analysis, the vehicle can decide upon a (potentially) fatal choice of action. Even if a system can detect obstacles in an image 99 percent of the time, it is almost guaranteed to make a mistake in a short time when it processes tens or hundreds of images per minute.

In contrast with the above description, consider a system that integrates visual information over time, constructing a visual map of its environment. Assume that a robot, again using range imagery, initially detects an object at a distance of 20 meters (the obstacle is actually a thin thistle bush). At this range, the system cannot be certain whether this object is a real obstacle; confirmation in subsequent images is required. By analyzing three or four new images of the scene, the program determines that the object is real, but poor sensor resolution permits only a crude estimate of the object's size and position. As the vehicle approaches the object, the increased resolution allows the robot to specify the size and position more precisely; again, agreement between estimates from one image to the next provides a high degree of confidence in the estimates. As the vehicle gets closer yet, the variance in the estimated depth of the pixels on the object indicates that the object is not solid, but more like an open, leafy, spine-like structure through which the laser can penetrate to different depths.

The next change in the description of the object is to a configuration of "sticks," a shape-description language (a specialization of generalized cylinders) well suited to describing thin objects such as fence posts and thistle bushes.

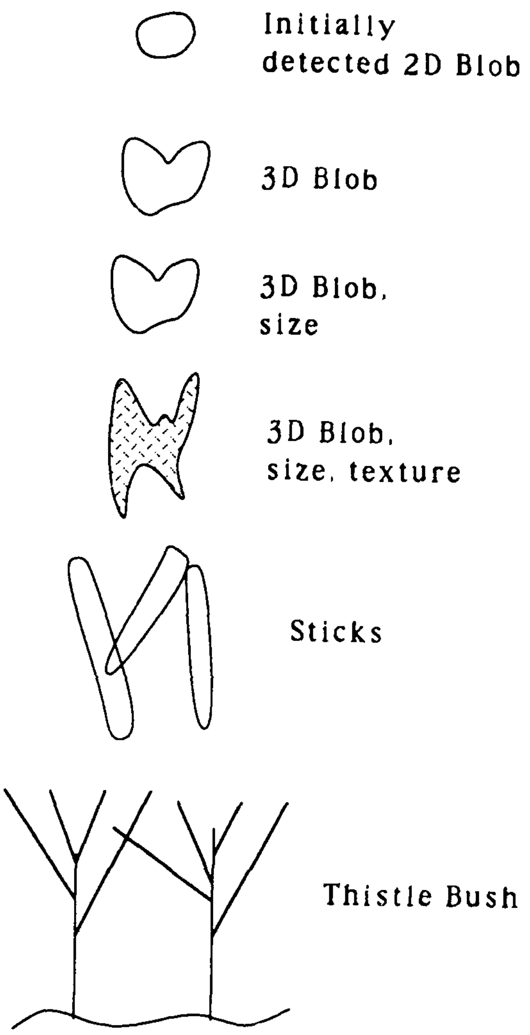


Figure 1: The evolution of the description of an object. As more information becomes available, and as more powerful representations can be computed reliably, the description of an object increases in its specificity.

And finally, because the descriptions of the object match those of thistle bushes, the robot can classify it as a thistle bush. This evolution of representation is shown in Figure 1. If, during the analysis that produces these descriptions, the bush is not detected in an image, the program does not assume that the bush has vanished. Rather, it attempts to explain why the bush was not detected. Perhaps it was out of the sensor's field of view, was occluded by another object, or was missed by the low-level segmentation process.

This simple description of a system that integrates information over time leads to the following observations:

- The underlying representation used to describe an object may change significantly over time. These changes are required for the incorporation of new knowledge, which can support stronger inferences about the object.
- A change in the representation of an object should occur only after the representation has been judged to be reliable. Stability over time is an important method of assessing reliability.
- There is more to determining stability than counting the occurrences of a description or maintaining the variance of a parameter. *Explanations* are required for the phenomena that can interrupt the continuity observed in an image sequence, such as occlusion, limited field of view, and image-processing errors.

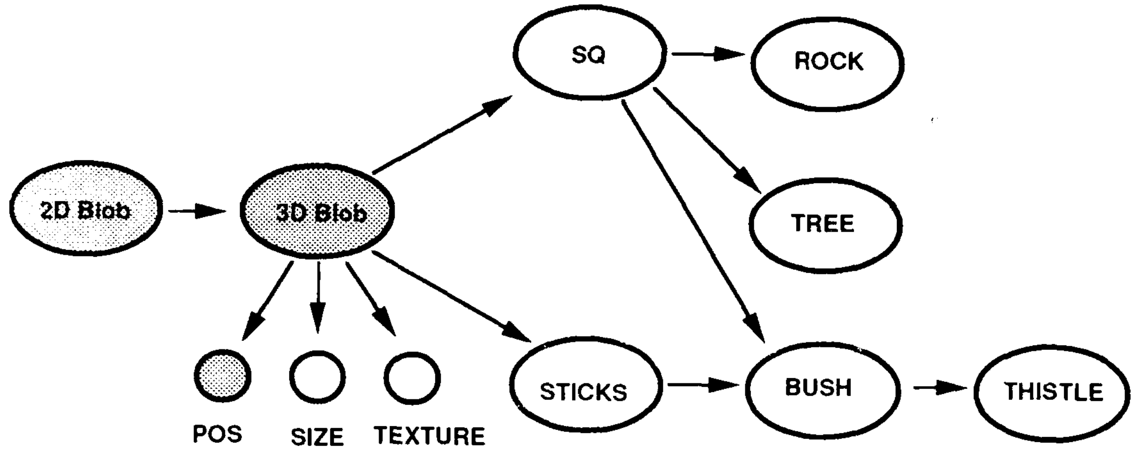


Figure 2: The representation space for the TraX system. The shaded nodes represent components of the representation in use. A new node can be shaded only if one of its connecting nodes is shaded and the stability conditions necessary for its acceptance have been met.

In this paper we develop these observations into an approach to building reliable descriptions of objects. We assume that a robot is attempting to construct a model of its environment as it gathers visual data; an autonomous land vehicle is an example of such a robot. The goal of the perceptual system is to integrate information as it becomes available into coherent representations of the objects in the environment. To provide a control structure for the integration of information, we introduce the concept of ***representation space***: a lattice of representations that is traversed as new information about an object becomes available. We contrast this structure with other approaches to multilevel representations. The concept of ***stability*** of a description is critical to our formulation because it serves as the basis for traversing representation space, for evolving the description of an object. Finally, we describe how the stability analysis is combined with representation space to produce reliable object descriptions. We illustrate ·:e power of these ideas by presenting results from the TraX system, a system designed to construct and refine models of outdoor objects detected in sequences of range data.

Representation Space

Assume, as in the introduction, that we have a vehicle or robot navigating some cross-country terrain and constructing a visual map of its environment. Whenever a new obstacle is detected, a description of it must be integrated into the vehicle's model of the world. To accomplish this task, we need to select a representation. As illustrated by the example of a robot approaching a bush, one can describe obstacles as simple blobs, textured blobs, or more detailed three-dimensional models, for example, using superquadrics or generalized cylinders. Which level or levels of description are appropriate?

The first important point of the thistle bush exampie is that different representations are appropriate at different stages in the analysis. Initially the bush can only be described as a blob with uncertain location. Eventually that representation can be improved to include size and texture information. Once the resolution of the data is high enough, a stable stick description can be computed. The importance of making these changes in representation is that as the representation of the object evolves, more powerful inferences about the object can be made. For example, when an object is represented as a blob, only rudimentary image properties such as position and spatial extent can be predicted. With a stick description, one can predict what the bush will look like from a different viewpoint, thus improving the performance of object-matching techniques; the structured shape description also supports the object-matching tasks required for localizing the robot in the world. For planning purposes, knowing that the object is a bush allows one to infer that the bush is not an important obstacle if the robot is a ten-ton ALV. By modifying the *type* of the representation over time, we can accommodate additional information and support additional inferences.

To make explicit the idea of multiple levels of representation, we introduce the concept of *representation space* —

a lattice of representations that is traversed as new information about an object becomes available. The importance of having a lattice is that it allows us to select representations that are appropriate for a given object, and the selection of the representations can be guided by characteristics of the object.

Figure 2 shows the representation space used in the TraX system. We consider representation space to be composed of *fundamental representations* and *enhancements*. Each fundamental representation reflects a qualitatively distinct representation, while an enhancement corresponds to the addition of a few parameters or an additional property.[1] In the diagram, each large node corresponds to a fundamental representation; each small node, to an enhancement. As indicated, fundamental representations available in our TraX system include 2-d blobs, 3-d blobs, superquadrics (SQ), sticks (a three-dimensional parts representation described later), and several semantic representations including BUSH and TREE.

Representation space is similar to scale space [8] in that the representation of an object is not restricted to any one level of description; different levels of specificity are possible. Unlike scale space, however, and unlike hierarchical representations [3,5] representation space is not homogeneous. For example, Marr and Nishihara propose using generalized cylinders of many scales to achieve a representation that spans data of different resolutions. Although the description of an object improves as more detailed information is acquired, there is no change in the type of inferences the representation can support. Only the level of accuracy improves. In representation space, however, a change in representation often implies the ability to assert new properties about an object.

One of the implications of representation space is that as new data are processed, the representation of an object can be modified in one of three ways. First, the parameters of the active components of the representation can be updated. We refer to this process as *refinement*; refinement procedures use standard filtering techniques and are similar to algorithms used by others to reduce parameter uncertainty [1,7]. The second type of change is the activation of a parameter or property attached to an active representation. For example, the active representation indicated in Figure 2 could be expanded by activating the TEXTURE node under 3D-BLOB. This type of modification is referred to as *enhancement*; the representation is enhanced by the addition of a new parameter. The final type of update is *augmentation*; in Figure 2 this would correspond to activating either the SQ (superquadric) or the STICK fundamental representation. The augmentation of a representation for an object means that the object can be described in a completely new vocabulary.

Arcs in the representation space diagram indicate ways that the representation of an object can be extended; that is, they provide the control structure for the evolution of the representation. A new node in representation space can become active — indicate that the corresponding representation is active for a given object — only if one of its connecting nodes is active. By shading nodes in this diagram we indicate the active components for a particular object. For example, in Figure 2 the large shaded node labeled 3D-BLOB indicates that a reliable three-dimensional blob description has been computed for the object. The small shaded nodes labeled SIZE and POS reflect the fact that the size and position of the blob are known.[2] Thus, for this particular object, the TEXTURE, SQ, and STICKS nodes can be activated the next time data for this object are analyzed.

How do we control the activation of representations. It is always possible to compute a stick or superquadric model or any other high-level representation of an object. However, a given description may not be a *valid* one. By valid we mean that the description correctly characterizes the object, as opposed to being a transient artifact of the algorithm. A description of a tree expressed in Euclidean solids may bear little relation to the actual structure of the tree. This observation underscores the second important point of the autonomous robot example of the introduction: stability over time is an indication that a description is valid. If a particular set of sticks is computed repeatedly, we assume that this consistency is caused by the fact that the sticks accurately model the shape of the object. The activation of representations and enhancements is the responsibility of supervisory processes that check for the stability of computed parameters. Examples of these processes are presented in the next section.

Note that the arcs in representation space do not imply computational dependency. For example, the algorithms in the TraX system for computing a superquadric model of an object are independent of those for computing a 3d-blob description. This differs from typical "levels of abstraction" hierarchies where each new description is computed from the previous level representation; such chaining of representations leads to the compounding of processing errors. In contrast, the different levels of representation space can be used to check the validity of a computed description. If the 3d blob predicted by the superquadric model is not similar to the blob computed directly, then the system

[1] We recognize that there is no formal distinction between levels and parameters. However, the intuition that there are several qualitatively different representations, each of which can be enhanced by the addition of a few parameters, is strong and we have found the distinction useful.

[2] For this discussion we are ignoring the issue of uncertainty in the estimate of a parameter. In actuality, once the measurement of a parameter is determined to be relatively stable we use Kalman filtering techniques to update the value of the parameter and maintain an explicit estimate of the uncertainty of the value.

would have evidence that at least one of its descriptions is not valid. While we have not yet explored this issue in detail, we hope to make use of the independence of representations to increase the overall robustness of the system.

Stability and Explanations

Representation space provides the basis for multiple levels of representation; the system associates a new representation with an object only after the representation has been declared valid. Currently, our principal indication of validity is stability over time, meaning that the bottom-up analysis of each image in the sequence produces essentially the same description of an object. This simple definition, however, is inadequate because errors in the single-image processing can alter these computed descriptions. Therefore we augment our definition of stability to allow for variations in description as long as the system can explain away these events by drawing on its knowledge of the task, the sensors, or the analysis procedures.

The technique we use for determining when a representation is stable is to construct a quasi finite-state machine (QFSM) for each component of the representation space. We use the term "quasi" because as an object moves through these states the history of its traversal is recorded and can sometimes affect the exact operation of the control structure.[3] Each QFSM is designed to incorporate the various pieces of knowledge relevant to that particular representation. Generically each QFSM has states labeled **Initially-Computed**, **Probably-Stable**, and **Stable** to represent degrees of stability. Other states that appear in the QFSMs include **Probably-a-Segmentation-Error**, **Stable-But-Missed-Once**, and **Unexplained**, which represent other assessments of the objects. The transitions between the states specify the reasons for changing evaluations. Reasons include **found-a-similar-representation** (in the new image), **did-not-find-a-match-but-can-explain-why-not**, and **no-match-and-no-explanation**.

Figure 3 shows a simplified version of the QFSM used for the analysis and tracking of 2-d blobs, the first representation invoked for an object. Thus, this control structure is responsible for determining when the 2d–blob representation should become active for a given object. Notice that there are several ways to enter the **Initially-Detected** state, including being detected in the first image of the sequence, coming out from behind another object, and splitting off a previously detected object. The importance of making these paths explicit is that we can later use the history to explain unexpected phenomena.

Once an object is **Initially-Detected** we try to match that object in subsequent images. As an object is successfully matched it moves into the **Stable** state; at this point the object is considered to be "real" and attempts to extend the representation are begun. If, however, after initial detection the object is no longer matched, the object quickly moves to the **Artifact** state, indicating that the detected obstacle is an artifact of some processing step and should be discarded.

Notice that at each state there is a **missed-but-can-explain** transition. This arc represents a situation where the object is not successfully located in an image in which it is expected, but there is an external explanation as to why not. These explanations are crucial because they provide a way to recover from apparent breaks in continuity, for which there are many reasons. Consequently, increasing the competence of the system requires recognizing these situations and incorporating explanations of them into the evaluation process. We currently have implemented the analysis required to support the following explanations:

- The object is no longer in the field of view of the sensor.
- The object is occluded by another known object.
- The object is a small, short blob far enough away that it can be easily missed.
- The object merged with another object to form a larger object.
- The object is unmatched because an error in the ambiguity interval assignment greatly changed the apparent characteristics of the object. (Ambiguity interval assignment is a preprocessing step necessary for determining the true range from a phase shift range image.)

In actuality, some explanations have a greater impact on the state of the object than simply causing them to remain in the same state. For example, if an object's absence is repeatedly explained by its having been merged into some larger object, then eventually that object is discarded; this removal of an object happens more quickly if the object entered the **Initially-Detected** state by splitting off of some other object.

[3] We could implement the control structure using a true FSM by simply increasing the number of states. We choose not to do so because we would end up with many states that were qualitatively similar, obscuring the general structure.

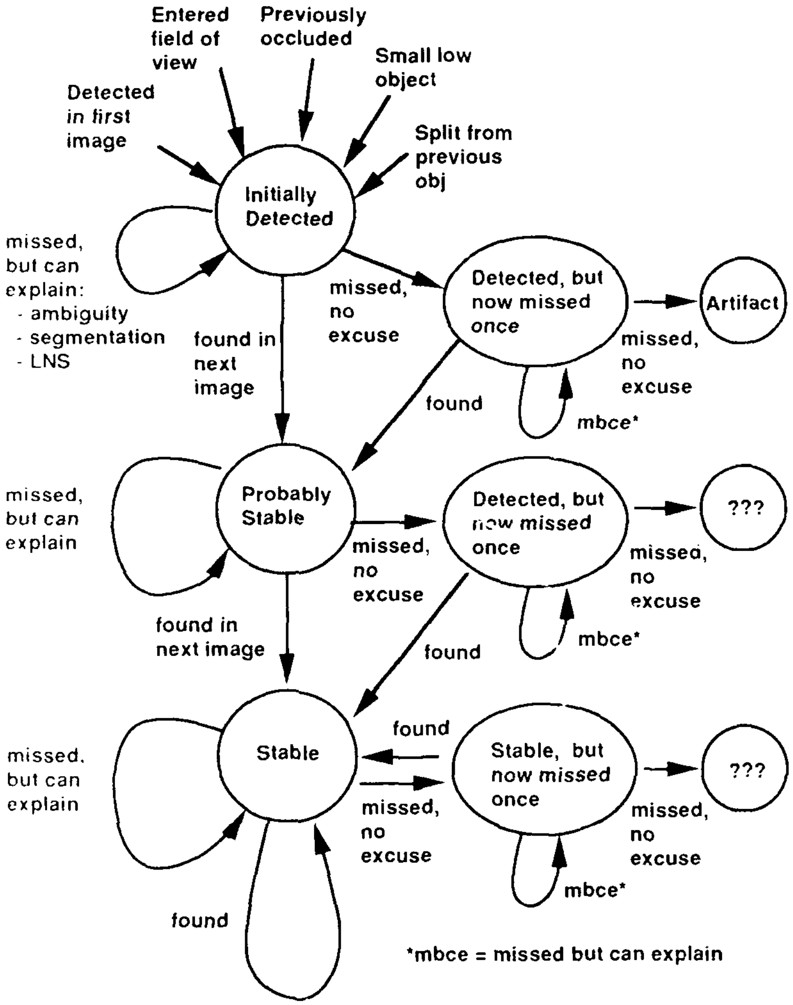


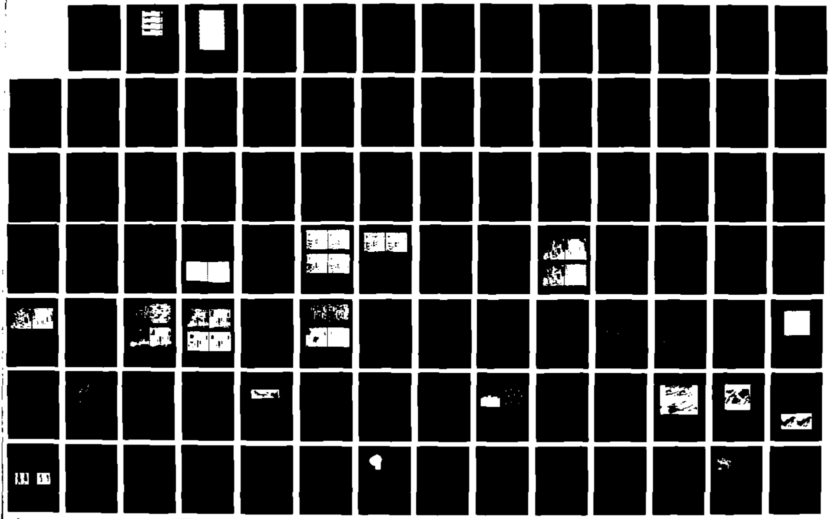
Figure 3: Part of the finite-state machine for determining stability of 2d-blobs.

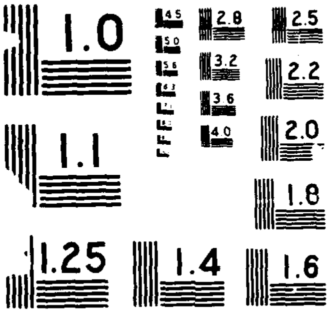
Figure 4 is a sequence of segmentation images produced by the single-image analysis. Object 1 (the short object on the right) is detected in all four images. Initially, this object is not matched in the fourth image because the object's shape has changed drastically. However, the change is mostly explained by the addition of a column of pixels in the last image. Because a column-scanning algorithm is used to disambiguate the range image generated by the phase-delay range finder, this change in an object's appearance is symptomatic of an ambiguity interval assignment error. Thus, the program concludes that an ambiguity error has probably occurred. Object 2 (the thistle bush to the left of object 1) is an example of a single object splitting (third image) and then merging again. In order to build a robust model of the environment the program must be able to handle situations such as these. The density of these events in this short sequence is higher than usual, but they are typical of the events that occur in our analysis of hundreds of images.

In the future we plan to expand the list of possible explanations. As we understand more of the fundamental properties of objects and more about the behavior of the analysis procedures we can implement more explanations, thus increasing the competence of the system.

Finite state machines monitoring the temporal stability of descriptions are used not only to activate a particular representation for an object, but also as a method for incrementally constructing the description within a particular representation. One example from our TraX system is the computation of "stick" models for objects. When objects are composed of thin pieces, as are fence posts and thistle bushes, the response of the range sensor tends to "fatten"

AD-A212 981   PROCEEDINGS OF IMAGE UNDERSTANDING WORKSHOP HELD IN   4/12
PALO ALTO CALIFORNIA O. (U) SCIENCE APPLICATIONS
INTERNATIONAL CORP MCLEAN VA   L S BAUMAN MAY 89
UNCLASSIFIED   N00014-86-C-0700   F/G 17/11   NL




1.0
4.5
5.0
5.6
2.8
2.5
3.2
2.2
3.6
4.0
2.0
1.1
1.8
1.25
1.4
1.6

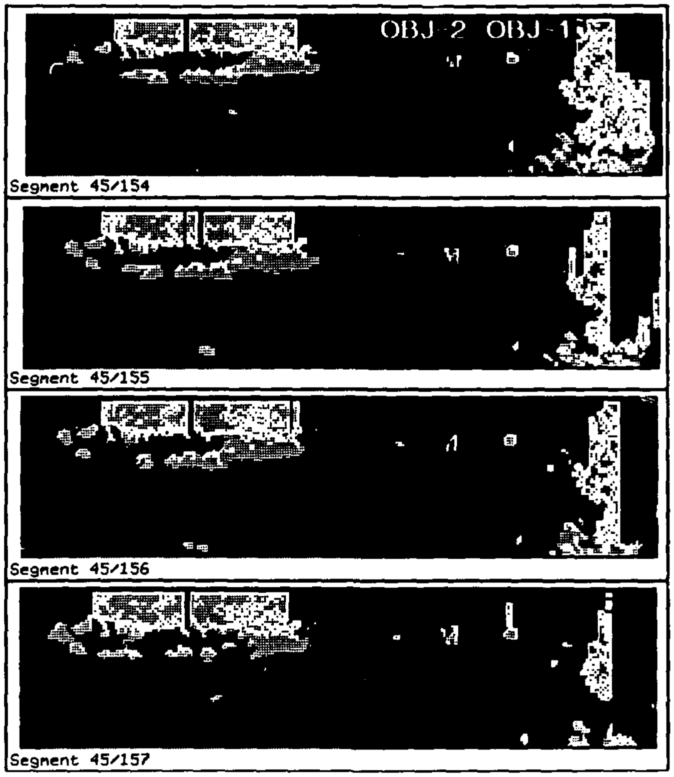


Figure 4: Tracking detected obstacles from image to image.

the parts by generating averaged range pixels along the sides. This blurring prevents other three-dimensional model constructing algorithms (e.g., fitting the data with superquadrics) from finding the true stick-like description. To model these thin objects we have designed a special representation we call *sticks*. By definition, sticks appear as one-pixel-wide lines in range images. Thus, to compute a stick model of an object, we first thin the range image of the object, and then compute a minimal covering; our technique for computing the covering is analogous to that proposed by Pentland and Bolles [6] for fitting superquadrics.

Figure 5 displays the results of applying the stick-fitting procedure to a detected object. Each model is computed independently, making no use of the previous solution. Note that most of the resulting models capture some structure of the bush. However, except for the last one, none captures all of the structure. The principal problem associated with minimal-covering modeling techniques is the lack of data to constrain the models. As a result, there are often many descriptions that characterize an object equally well. Although many approaches can be taken to improve the stick fitting procedure, our goal is to use temporal stability to compute a more robust model.

In the interest of brevity we omit a detailed description of the strategy used for determining a stable stick representation. The basic approach is to attempt to match sticks in the current model of an object to sticks computed from a new range image. Model sticks that are matched by the new sticks are reinforced in terms of their stability, and their parameters are updated using standard Kalman filtering techniques. (We use our model of the sensor and its noise characteristics to estimate the variance of the new measurements.) Sticks that are detected once but not matched again are eventually discarded with the explanation that they were an artifact of the stick fitting procedure. Figure 6 shows an example of the stability analysis applied to sticks. On the left is the stick description computed independently using the single range image as input. On the right is the set of stable sticks tracked over time. A new stick is added to the model on the right only after it has been deemed stable. Note that the stable description converges to (what is known to be) an accurate model of the bush.

To underscore the role of explanations in the stability analysis, consider the case in which a model stick has been stable for some time but is then missed repeatedly in new images. Currently, we include only one explanation that allows this stick to remain as a viable part of the representation: the vehicle has backed away from the object

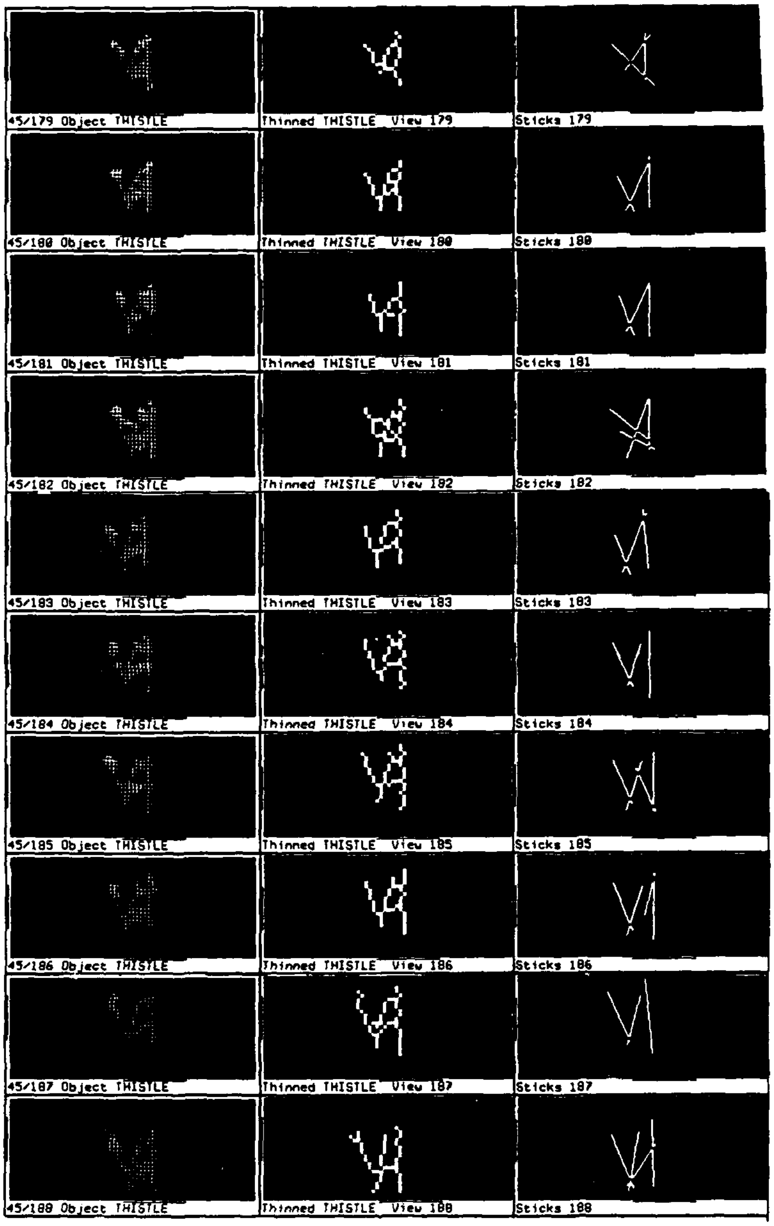


Figure 5: Computing stick descriptions of a thistle bush. The column on the left displays the silhouette of the object as determined by the obstacle-detection procedure; the middle column is the thinned version of the these objects. The right column displays the best "stick" model of the thinned bush. Note that some of the sticks are quite robust, such as the vertical stick on the right. Others are less stable, while some are artifacts. Though the fitting technique can be improved, our goal is to use temporal stability to compute a more robust model.

and the stick may no longer be detectable by the sensor. If this is not the case, then the stick is marked as an **unexplained-disappearance**. We plan to extend the stick QFSM to enable the system to explain a wider range of possibilities, including situations in which two new stable sticks replace an old one.

Summary

Our goal is to design a representation scheme that provides a natural mechanism for a representation to evolve over time. In this paper we propose the concept of a representation space that provides a graduated set of object

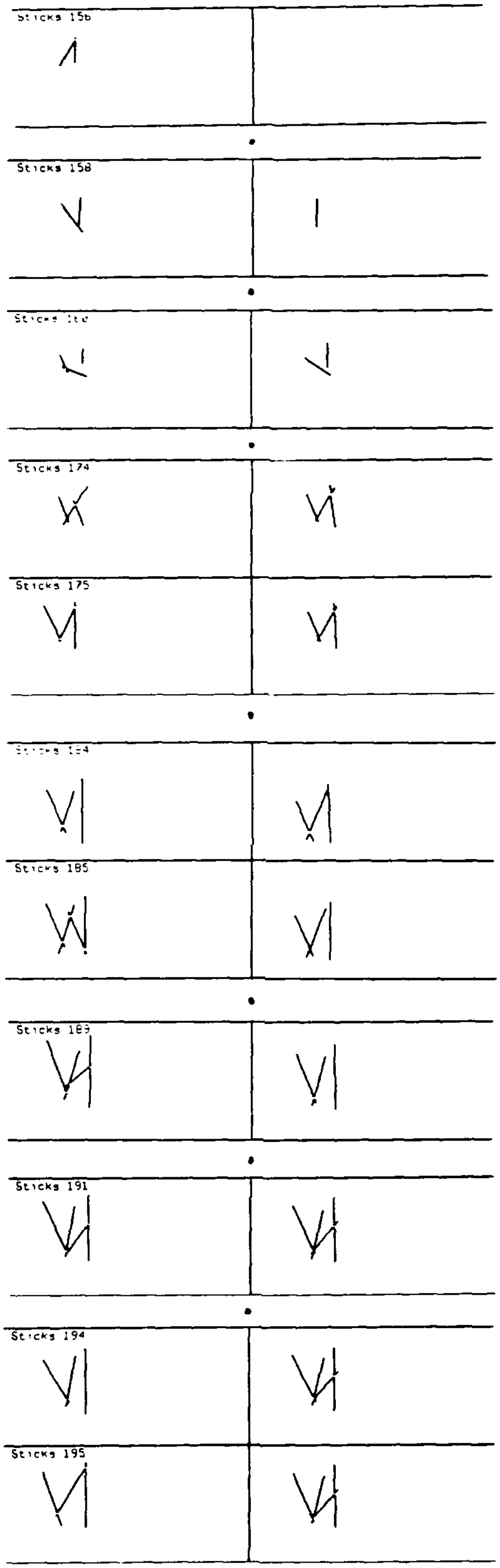


Figure 6: Sticks computed independently (left column) and tracked over time (right column). As a stick becomes stable it is added to the model on the right.

descriptions. Given these multiple descriptions, which ones are appropriate for a specific object? To answer this question we introduce a definition of stability that incorporates explanations to account for the problems and special cases that invariably arise in the processing of real imagery. In this paper, stability is our key to reliability.

As part of the idea of a representation space we defined three types of changes to an object's representation: refinement, enhancement, and augmentation. Refinement updates parameters that are currently active; enhancement activates new parameters for a currently active representation; augmentation activates completely new representations. Figure 1 illustrates the operation of these updating procedures as they transform an initially detected blob into a semantically classified object. The figure shows the changes in representation that occur as the robot approaches the aforementioned thistle bush. The bush, when it is initially detected, is represented as a two-dimensional, image-based blob. After additional images have been processed and the three-dimensional location of the blob stabilizes, the object's description transforms into a 3-d blob. When the size parameter of the 3-d blob is stable, the 3-d blob is enhanced to include a size parameter. When the texture parameter stabilizes, the 3-d blob is enhanced again, this time by a texture parameter. When a stick description for part of the bush is deemed to be stable, the bush's stick representation is activated. As more sticks stabilize they are added to this description level. And finally, after enough sticks have stabilized to form a thistle-like structure, the bush's representation transforms into a thistle bush.

This ability to change an object's representation incrementally is crucial in autonomous navigation tasks because objects are viewed many times, from different viewpoints, and with different resolutions. By continually updating the objects' descriptions, an autonomous robot is in a position to base its decisions on the most current information at all levels, including the semantic level.

In the future we plan to continue exploring the idea of using stability to evaluate the reliability of representations. In particular, we plan to (1) implement finite-state machines for analyzing the stability of superquadrics and other representations, (2) develop new explanations based on support and gravity, and (3) explore ways to combine other types of reliability with stability.

Acknowledgments

We gratefully acknowledge the comments of the SRI vision group and the assistance of Tom Strat, Andy Hanson, Steve Barnard, Helen Wolf, and Lynn Quam. The range data were gathered at Martin Marietta in Denver Colorado with the assistance of Steve Seida, Jim Allison, Terry Dunlay, and Laura Haber. This research was supported in part by DARPA contracts MDA 903-86-C-0084 and DACA 76-85-C-0004. This paper will appear in the Proceedings of Computer Vision and Pattern Recognition, San Diego, June 1989.

References

1. Ayache, N., and O. D. Faugeras, " Maintaining Representations of the Environment of a Mobile Robot," *Proc. of International Symposium on Robotics Research,* Santa Cruz, CA, 1987.

2. Daily, M. J., J. G. Harris, and K. Reiser, "An Operational Perception System fo [illegible] Country Navigation," DARPA Image Understanding Workshop, Washington, DC, 1988.

3. Marr, D., and H. K. Nishihara, "Representation and recognition of the spatial organization of three-dimensional shapes," *Proc. R. Soc. Lond. B,* **200**, 269–294, 1978.

4. Matthies, L., and T. Kanade, "The Cycle of Uncertainty and Constraint in Robot Perception," *Proc. of International Symposium on Robotics Research*, Santa Cruz, CA, 1987.

5. Nishihara, H. K., "Intensity, Visible-Surface, and Volumetric Representations," *J. of Artificial Intelligence,* **17**, 265 284, 1981.

6. Pentland, A. P., and R. C. Bolles, "Learning and Recognition in Natural Environments," to appear in the *Proceedings of the SDF Benchmark Symposium on Robotics Research*, MIT Press, 1988.

7. Smith, R., M. Self, and P. Cheeseman, "A Stochastic Map for Uncertain Spatial Relationships," *Proc. of International Symposium on Robotics Research*, Santa Cruz, CA, 1987.

8. Witkin, A., "Scale Space Filtering," *Proceedings of IJCAI,* pp. 1017 – 1022, Karlsruhe, 1983.

Carnegie Mellon Navlab Vision

Charles Thorpe and Takeo Kanade
Robotics Institute and School of Computer Science
Carnegie Mellon University
Pittsburgh PA 15213

Abstract

In the four years of the Carnegie Mellon Navlab Vision project, we have built perception modules for following roads, detecting obstacles, mapping terrain, and recognizing objects. Together with our sister "Integration" contract, we have built systems that drive mobile robots along roads and cross country, and have gained valuable insights into viable approaches for outdoor mobile robot research. This work is briefly summarized in the first part of this report.

Specifically in 1988, we completed one color vision system for finding roads, began two others that handle difficult lighting and structured roads and highways, and built a road-following system that uses active scanning with a laser rangefinder. We used 3-D information to build elevation maps for cross-country path planning, and used maps to retraverse a route. Progress in 1988 on these projects is described in the second part of this report.

Introduction

This report consists of an overview of accomplishments during the four years of our Navlab Vision project; discussion of progress during 1988; a compendium of our insights and practical advice for building mobile robots; a discussion of future directions; and selected publications of our research group.

Overview of Accomplishments

Outdoor mobile robot vision research at CMU has been funded by DARPA since January 1985. The scope of the project is very broad and has included cross-country runs and obstacle detection as well as road following; direct 3-D sensors along with video cameras; object recognition and terrain mapping; and close cooperation with the Warp group and with the Navlab Integration work, to build complete mobile robot systems. Progress for each of the four years is described in the year end contract reports [16, 34, 35, 36].

Color-based Road Following. The culmination of our road-following work is a reliable system that drives the Navlab along a narrow, twisting, tree-lined bicycle path. The heart of the system uses adaptive color classification, which automatically adjusts for changes in road appearance or lighting conditions. Variants of the system use two cameras to extend the dynamic range to handle deep shadows; find intersections of known shape; incorporate additional features such as texture; and use the Warp processor for high speed. The latest version uses the Warp to achieve a 2 second processing loop, allowing vehicle speeds of 1 meter / second even on our narrow test course.

Terrain Representation and Obstacle Detection. We have developed three levels of terrain representation corresponding to different resolutions at which the terrain is described [11, 12, 13, 19]. At the low resolution level we describe only discrete obstacles without explicitly describing the local shape of the terrain. We used this level for fast obstacle detection and avoidance. At the medium level, we include a description of the terrain through surface patches that correspond to significant terrain features. At that level, the resolution is the resolution of the operator used to detect these features. This level of representation is especially useful for cross-country navigation in which we have to deal not only with large discrete obstacles but also with the changing shape of the terrain. This representation has been successfully demonstrated in conjunction with a path planner developed under the Integration contract. Finally, the description with the highest resolution is a dense elevation map whose resolution is limited only by the sensor. The techniques we developed for this representation provide a complete description of the terrain including occluded regions and uncertainty. After the low-resolution obstacle detection was demonstrated as part of the Navlab, it was ported to Martin Marietta. Work in conjunction with Martin reduced run time to less than one half second, the frame rate of the ERIM scanner. This was the only project during the Martin ALV contract that was developed outside of Martin, integrated into the ALV, and used in one of the ALV main demos.

Map Building and Matching. In addition to extracting snapshot maps of the terrain from range images, we have developed algorithms for matching and merging individual maps into a single consistent representation. Again, the matching algorithms are applied to the three levels of representation: At the lowest level discrete obstacles are matched in order to compute the displacement between consecutive maps. At the medium level terrain features are matched to compute the best consistent match between maps. At the highest resolution maps are directly correlated to compute the displacement by a minimization technique. The accuracy of the resulting displacement can be as good as the resolution of the map (as good as 10 cms in our experiments) [13, 19].

Road Following by Active Sensing. Our ERIM scanner measures not only distance to each point but also reflectance. If the road surface (e.g. asphalt) has much different reflectance than the surroundings (e.g. grass), it is straightforward to detect and track the road. For situations in which reflectances do not significantly differ, such as dirt shoulders, we have to pay attention to details of signal attenuation, grazing angle, and surface fitting in order to find the road border. Since the ERIM uses its own laser as its light source, it is insensitive to shadows or lighting changes. This system has even driven the Navlab at night. This method has also been ported to Martin Marietta, and has driven the ALV [12].

Systems. The Road Following vision modules has been integrated into the systems built by our Integration work. Highlights of these systems include:

- Navigating the Schenley Park bicycle path, starting with a crude map and producing an updated map. This system included color vision for road-following; range data analysis for mapping both discrete obstacles (trees) and terrain; intersection recognition and navigation; a planner that followed the road and avoided obstacles; and sequencing to predict road appearance and to tell perception when to take an image. The system was based on CODGER, our adaptation of blackboard ideas for mobile robot navigation developed as part of our Integration research [5, 6, 25, 27, 28].
- Navigating the CMU sidewalk network, using a preloaded map to predict object appearance and to choose between a forward-looking and an angle-mounted camera to see the next sidewalk or intersection. The map was also used to invoke a program to locate stairs, which used a "colored-range image" built by fusing camera data with xyz data from the rangefinder [4, 20, 37, 38].

Sonar. Some of our earliest successful outdoor runs used Moravec and Elfes's sonar system, originally developed for indoor use, to drive our Terregator robot in Schenley Park. The sonar system was very good at mapping and avoiding natural obstacles such as trees [3, 23].

Stereo. The FIDO stereo system was ported from indoor laboratory robots to the Terregator, and reimplemented on a prototype Warp. It successfully steered the Terregator around man-made outdoor obstacles, but was less successful with trees and bushes. Future systems could use the complementary strengths of sonar and stereo to build complete and reliable mapping [21, 22, 25, 29].

Other Road Detection Methods. Our early systems tracked edges, oriented edges, road cross-section profiles, correlation winJow outputs, and other features [37, 38]. Each of these methods works well, but only in particular circumstances. Current research is using several of these operators together to track the lines, shoulders, and other features of public highways [17]. A model-based control program will take advantage of the structure of highways to decide which features to track and how to track them. This approach should be robust as well as efficient. Other current work is exploring new methods, such as an unsupervised color classification scheme that uses shape information but does not need color data from previous images. This scheme is not susceptible to quickly changing illumination, and can find the road at the beginning of a run to initialize the color tracker [1].

Calibration. Our multi-sensor perception experiments need to know the geometrical relationship between sensors. Even for a single sensor, it is important to know the transform from sensor to vehicle coordinates. Our best calibration system uses images of two grids of points to build transform lookup tables, or to derive traditional camera parameters such as location, piercing point, row and column vectors, etc [7].

Object recognition. In order for a mobile robot to perform a meaningful mission, it must be able to see and recognize known objects. Examples of our object recognition work are two programs for recognizing cars, one using color data and the other using range images. Color car recognition used hierarchical grouping, in which edges are grouped into lines; lines into parallels; parallels into trapezoids; and trapezoids into connected sets that could be car roofs, windows, trunks, or hoods. Starting with range data, the 3-D system first detected flat surfaces, then applied single-surface constraints such as range of orientations allowed for a roof or door, then used surface-pair constraints such as the angle between a roof and door. Both methods work on several views of different cars [11, 12, 20].

1988 Progress

In 1988 we neared completion of one of our road following programs, and began work on three new road followers. Our range data processing built maps and, in conjunction with NASA sponsorship, began very high resolution terrain analysis. The highlights of these projects, and of the systems that use them, are briefly described below.

SCARF: In 1988 we completed SCARF, our system for Supervised Classification Applied to Road Following [1]. SCARF is the logical continuation of a long chain of road following programs that use color classification. The first implementation of SCARF in 1986 ran on Sun workstations, with 32 by 30 pixel images, in about 12 seconds per image. Later implementations of that version ran on the prototype Warp and on production Warps, with speeds as fast as one image per 4 seconds.

Over the past year and a half, we have upgraded SCARF to use, first, higher resolution images (60 by 64), and, second, two images to increase dynamic range. This slowed our runs to tens of seconds per image, even on a Warp.

Now, taking advantage of compiler upgrades for the Warp's W2 language, and doing some code restructuring, we have reimplemented SCARF on the Warp. Our processing time is now down to 2 seconds per image. We moved almost all of the code onto the Warp cells themselves. Further, we reduced the number of calls to the Warp per image from 14 (last year) to 3 (earlier this year) to 1 (now). After initialization, we pass the Warp cells each new image, and get back only the new road location. All of the system state is saved on the cells from call to call. We also have debugging versions that can extract classification information for display, but those extra Warp calls and data movement slow down the system. Current running time is 1 second of Warp time per image.

The full formulation of the probability equation used in classification includes the log of the determinant of each class. Early implementations of SCARF on the Warp have always avoided logarithms, since there is no log function in W2. On benign data, this did not cause any problems. But running with the Navlab outside on a snowy day, the system did not work correctly. In our standard test sequences, each class had approximately the same size determinant (i.e., the classes had approximately equal variance), so we could safely ignore that term. But on a snowy day, the "snow" and "road" classes each had very small variance, while the "trees + parked cars + trash barrel" class had a much larger variance. This imbalance caused improper classifications. We worked with the Warp group to include a log macro and to compile it into our W2 code. The resulting system performs no better on most of our images, but dramatically improves performance on snowy days and under similar circumstances.

The resulting system has driven the Navlab many times, along our narrow bicycle path in Schenley Park. The top speed at which we have run is one meter per second, the length of our test course (compared with 20 cm/sec last year). With the fast processing loop and the complete formulation of probabilities, the vision results are solid. While vehicle speed has always been a secondary concern of our work, we can now drive at moderate speeds on our difficult test course, and should be able to use the same system to drive at higher speeds on wider, straighter roads.

UNSCARF: One of our new road detection algorithms for this past year is UNSCARF, for UNSupervised Classification Applied to Road Following [1]. A large problem with our early road perception work was dealing with rapidly changing illumination. If the sun is covered by a cloud, the lighting is diffuse and we can follow roads with a single camera. If the sun is out, there are problems with camera dynamic range, but our methods that use two cameras work. But if the sun is quickly covered or uncovered by clouds, then colors change and shadows change and the brightness changes. If object appearance differs greatly between successive processed frames, current methods have a hard time tracking the road.

UNSCARF places less emphasis on colors and more on shapes. Instead of classifying each pixel according to statistics from previous images, it groups neighboring pixels using unsupervised clustering methods. We have found that clustering with 5 parameters (R,G,B and row,col) gives us classes that are both homogeneous in color and connected in the image. We then piece a road shape together out of those clusters, instead of from individual pixels. Evaluating candidate roads uses shape cues such as parallel edges, smooth edges, edges the right distance apart, and so forth. The combination of unsupervised classification and evaluation with shape cues makes UNSCARF tolerant of the large illumination changes that have given problems to previous systems.

FERMI: FERMI deals with public highways and roads, that have more structure and variation than our Schenley Park test site [17]. The key to handling diverse roads is explicit modeling of the colors, shapes, and features of each road type. FERMI has a representation that lists width, maximum curvature, color, surface type, location of lines, type of shoulders, presence of guard rails, type of adjacent vegetation or soil, illumination conditions (sunny or cloudy), illumination direction, and so forth. By having many simple experts, one for tracking each type of feature, we are able to follow many kinds of roads within the same control framework. None of the individual trackers (edges, lines, color discontinuities, etc.) that we explored in our early work were adequate by themselves for road following. But by incorporating many of them into a single system, and intelligently selecting which tracker to use to follow which feature, we expect FERMI to be reliable and flexible. In 1988, FERMI has been designed and partially constructed, and has driven the Navlab.

ERIM Reflectance: A new project for 1988 is road tracking using the ERIM reflectance data. Our ERIM laser rangefinder produces not only range at each point but also magnitude of reflectance. Since the scanner produces its own illumination, the reflectance images are not distorted by shadows or sunlight or changing cloud cover. Reflectance is affected by distance (less of the illumination is reflected back to the scanner from more distant objects), but this can be compensated for by using the range data. Thus many of the sources of error in standard video images are not present in active reflectance data.

There still are, however, some problems with using reflectance data. The magnitude of the reflectance changes with grazing angle: the road at larger distances appears at a shallower angle, and reflects less. Reflectance also changes from place to place along the road, as the road surface goes from dirty to clean or from wet to dry. And finally, since reflectance is only a single channel (rather than the three channels of an RGB camera), not all objects have distinct appearances.

The solution to the grazing angle is to process each image as a series of horizontal bands, so within each band the grazing angle is approximately constant. We keep separate appearance statistics for each of the bands. We handle changes from place to place by updating our appearance models each image. The problem of multiple objects with the same appearance is more difficult. Part of the solution is to limit processing to a region around the predicted road location. Another answer is to use geometric constraints, such as expecting road edges to be locally parallel. But the effectiveness of these solutions depends on the materials that form the road and its borders. Asphalt and grass have much different reflectances, so the portion of our test path that is grass-lined is easy to segment. Dirt, however, can appear much more like asphalt, so in dirt-lined segments we have to use more detailed processing, such as tracking a single road edge when the other edge is indistinct.

Our program to follow roads using ERIM reflectance has run the Navlab many times, including runs at night. This is the first time we have had a usable day/night road following system. The program was also transferred to Martin Marietta, and successfully drove the ALV.

In addition, this work provides the first step in a new project in building and re-using maps. As we drive, we record the position of the road (from reflectance analysis) and of obstacles (from range analysis). When we later retraverse the same path, we use the detected positions of the road and obstacles to locate the Navlab on the map. The map can then be used to predict upcoming obstacles or turns in the road, and to plan paths beyond the current field of view.

Our work with reflectance processing and road mapping is described in [12, 14].

Terrain Mapping: Algorithms that build a terrain description made of polygonal regions have been implemented and demonstrated on the Navlab. The resulting description is a mesh of polygons built from an Erim image, each of which is a feature of the terrain. This terrain modeling program provides the type of information required by the new path planner [28]. The combination of terrain modeling and path planning has been demonstrated on the Navlab and is a major step toward cross-country navigation and the implementation of the Core system.

Terrain mapping work is included as part of an overview of all our range data analysis research in the past four years in [12].

Insights and Advice

Through the course of our work, we have developed some basic tenets of developing outdoor mobile robots. While some of these are scientific insights, most of them have the flavor of pragmatic advice. The most important include:

- Computing is a bottleneck. Our best results use the Warp, rather than a Sun, to gain processing speed. The extra computer power is mostly used not to drive the robot faster but to process images more frequently. Processing images more frequently in space means easier predictions, more objects shared between successive images, and smaller changes in apparent size and shape. Processing more quickly in time means less sensitivity to lighting changes. The 100 MFlops of the Warp help give us a 2-second loop for our current color vision algorithm. But processing remains a bottleneck. Even for the same algorithm, we could use an additional factor of 60 to get to frame rate, times an additional factor of 64 to process higher-resolution images.

- Development environments are a bottleneck. While the Warp gives us vast improvements in processing, until recently it was difficult to harness that power. Hardware developers and computer engineers tend to expect their users to have a few well-specified algorithms that can be compiled once and run many times. But it is the nature of research that programs and parameters need to be changed frequently. To be useful, a supercomputer needs to have debuggers, hardware diagnostics, easy access to display devices, and compilers that run in reasonable amounts of time. Fortunately, those are now becoming available on the Warp.

- Simplicity helps. Object models, algorithms, and systems should be no more complex than needed. A road model, for instance, that attempts to derive too many geometric parameters from a single interpreted image, may be subject to large instabilities due to small errors. We have had much greater success in modeling our road as locally planar and straight. By solving only for two parameters (the road's heading and lateral offset from the vehicle), we have a stable solution insensitive to minor noise. And by processing quickly, we can track the road as it does eventually turn or pitch, and compensate as we arrive at those points.

- 2-D is often adequate. We often create a distinction between Image Understanding (IU) (3-D, reasoning about the scene, physical models) and Pattern Recognition (2-D, reasoning about the image, statistical models). In the IU community, the usual assumption is that IU methods are always better. But for many of our road following experiments, we do not have the detailed object and illumination models needed for a real IU approach. Instead, careful attention to the details of pattern recognition approaches (supervised and unsupervised clustering in color space, Bayesian classificatiun, Hough transforms in image coordinates, etc.) allows us to build functioning systems. One reason for the success of pattern recognition methods is that many of them are suitable for parallel implementation. The high speed of our Warp programs allows us to process more closely-spaced images, and compensates for the relatively weak scene models.

- Direct sensing helps. Reasoning in 3-D is much easier when the data starts out in 3-D, such as from a scanning laser rangefinder. Our ERIM data is not perfect, but gives us an excellent starting point for obstacle detection, terrain mapping, and 3-D object recognition.

- Image Understanding (IU) is still needed. There is no direct sensor for "road" or "tree". Furthermore, there are objects and tasks that we do not yet understand how to handle with simple algorithms and models. So even with good 3-D and color sensing, it is still necess.ry to do all the IU tasks of modeling and interpretation. Direct sensing may eliminate some of the messy low-level interpretation, but does not eliminate the need for fundamental work in IU. Pattern recognition often works, but does not use the physical 3-D models that are needed for interpreting complex scenes.

- The world changes. Our early outdoor stereo work was foiled by wind-blown trees. Early color vision made assumptions about constant appearance, and ran afoul of variations in grass color from place to place. Fairly sophisticated vision systems can be fooled by a cloud suddenly covering the sun, which changes not only the intensity but also the color of illumination. The appearance of the road changes from one run to the next, due to our own tire tracks, oil drops, and other effects.

- Sensors are a bottleneck. Too much effort goes into overcoming insufficient dynamic range, fighting noise, and modeling errors. Our solutions include using 2 cameras mounted very close to each other, with different iris settings, to extend the dynamic range. This is an engineering solution to a technology problem, and diverts effort from science. Yet this sort of "hack" is needed to use many current sensors.
- Integration is difficult but crucial. Capable mobile robots need multiple sensors, probably with multiple sensor interpretation methods, and have multiple goals and multiple control schemes. If the individual components are designed separately, they are not likely to work together. Much of our design and testing effort has been devoted to working with our sister Integration effort to build systems that can follow roads and avoid obstacles; that can look for landmarks while looking for roads; and that can handle other conflicting demands.
- Easy tasks are easier than expected, hard tasks are harder than expected. Following a well-lit sidewalk, bordered by green grass, is nearly trivial. Following a winding path with dirty asphalt, bordered by trees, grass, dirt, and fallen leaves, with changing lighting, is much more difficult.
- Do not trust laboratory simulations, or runs on a few canned images. Simplified or reduced test data is useful for first debugging, but success in the lab does not guarantee success outdoors. There is no substitute for lots of experimental runs.
- Mobile robot research is increasingly important. Results from our work have already been directly applied to interpreting sonar data (for design studies of an underwater autonomous vehicle) and to mapping terrain for planning footfalls for a walking planetary rover. The ideas and experience coming from our project have influenced many other mobile robots, ranging from underground mining vehicles to other road following efforts. And in general, the Navlab Vision work is part of a paradigm shift in image understanding research, moving from generic interpretation of single frames of laboratory data to goal-driven analysis of streams of images from a real, continuously moving, outdoor robot.
- Technology transfer is difficult but crucial. Success in our work will ultimately be measured by its impact outside the research community. We have ha. some success in transferring both code (e.g. the CODGER blackboard) and ideas (via papers). But by far the most effective technology transfer is by people exchange. Our industrial visitors and graduate students have not only learned particular algorithms, but have also absorbed the Image Understanding culture and methods of attacking problems.

Future Directions

Our work is expanding from simple systems to more integrated approaches, and is beginning to include other research directions such as machine learning.

Multiple approaches will be needed for general robots. Where simple approaches and 2-D techniques work, we can build simple and therefore reliable systems. But where those methods fail, we need to resort to more time-consuming and complex (and potentially buggier) methods. In path planning, for instance, we have both a simple pure pursuit tracker for following unobstructed gently curving roads, and a path planner that uses a complete kinematic vehicle model for avoiding obstacles and cross-country traverse.

Our systems to date have not used multiple approaches. Instead, each system had a single module for each function, so we could test that particular approach. We now understand basic navigation to the point where we can build a reliable navigation platform, and do research on applications that assume that navigation is provided. Building the basic platform will require providing a repertoire of perception and planning methods, all of which work through a common representation.

The unifying representation for our next systems will be the Local Environment Map, or LEM. It describes the immediate vicinity of the robot in terms of terrain shape and type, and includes symbolic labels such as "mud", "road lane boundary", or "rough terrain". The LEM is built by 3-D sensing and color vision, with contributions from specialized experts for particular situations, such as terrain typing, road tracking, or high-resolution mapping. The LEM is updated as the vehicle moves anu iooks at the terrain from new directions. Finally, local trajectory planners will use the LEM for choosing the vehicle path.

A new research direction we are beginning to pursue is the incorporation of learning. Our FERMI system uses models of road features, vehicle motion, and the capabilities of individual feature trackers to follow structured roads. This style of model-based vision is an excellent candidate for explanation-based learning. The system needs to detect its mistakes, explain why they were committed, and update its models. If, for instance, the double yellow line in the center of the road is not found at its expected location in an image, there are several possible explanations and corresponding model updates:

- If other features are found, but shifted from their expectations, perhaps the road turned and the geometric model should be updated.
- If the tracker failed because the line entered a shadow, a different tracker should be selected and the lighting model should be changed for all trackers in the next image.
- If the double yellow line ended, there may be an intersection, which will change expectations for road edges as well.

The power of model-based road following is that we have enough redundant information to detect and diagnose failures. The power of explanation-based learning is that we can use these diagnoses to update our models, and improve our understar.ding of scene geometry, lighting, vehicle motion, and operator performance.

Acknowledgements

Many researchers have contributed greatly to this research. Faculty include Martial Hebert, Katsushi Ikeuchi, Takeo Kanade, Eric Krotkov, Hans Moravec, Steve Shafer, Chuck Thorpe, Jon Webb, and William Whittaker. Students are Omead Amidi, Alberto Elfes, Jennie Kay, Karl Kluge, InSo Kweon, Dean Pomerleau, Doug Reece, Bruno Serey, Tony Stentz, and Rich Wallace. Staff who have contributed directly over the four years include Paul Allen, Mike Blackwell, Tom Chen, Jill Crisman, Kevin Dowling, Thad Druffel, Eric Hoffman, Ralph Hyre, Bala Kumar, Jim Moody, Tom Palmeri, Richard Redpath, Jean-Christophe Robert, David Simon, Jeff Singh, Hans Thomas, and Eddie Wyatt. And the visiting researchers who have worked with us are Arun Aggarwal, Taka Fujimori, Yoshi Goto, Keith Gremban, Hide Kuga, Kichie Matsuzaki, Taka Obatake, and Masatoshi Okutomi.

The research described in this report is sponsored by the Strategic Computing Initiative of DARPA, DoD, through ARPA Order 5351, and monitored by the US Army Engineer Topographic Laboratories under contract DACA76-85-C-0003, titled "Road Following." Portions of this research were also partially sponsored by the National Science Foundation contract DCR-8604199, by the Digital Equipment Corporation External Research Program, and by NASA under contract NAGW-1175.

Selected Publications

[1] J. Crisman and C. Thorpe. Color Vision for Road Following. In *SPIE Conference on Mobile Robots*. November, 1988. A different version appeared in the proceedings of SIMAP 88, University of Osaka, Japan, May 1988.

[2] J. M. Cuschieri and M. Hebert. Sonar Applications for Underwater Vision. In *ASME Symposium on Current Practices and new Technologies in Ocean Engineering*. ASME, January, 1988.

[3] A. Elfes. Sonar-based real-world mapping and navigation. *Journal of Robotics and Automation, Vol. 3* , 1987.

[4] Y. Goto, K. Matsuzaki, I. Kweon, and T. Obatake. CMU sidewalk navigation system. In *Fall Joint Computer Conference.* ACM/IEEE, 1986.

[5] Y. Goto and A. Stentz. Mobile Robot Navigation: The CMU System. *IEEE Expert* , 1987.

[6] Y. Goto, S. A. Shafer, and A. Stentz. *The Driving Pipeline: A Driving Control Scheme for Mobile Robotics.* Technical Report CMU-RI-TR-88-8, Carnegie Mellon University, The Robotics Institute, June, 1988.

[7] K.D. Gremban, C.E. Thorpe, and T. Kanade. Geometric Camera Calibration using Systems of Linear Equations. In *1988 IEEE International Conference on Robotics and Automation.* April, 1988. Also in 1988 Proc. of Image Understanding Workshop.

[8] M. Hebert and T. Kanade. First results on outdoor scene analysis. In *Proc. IEEE Robotics and Automation.* San Francisco, 1985.

[9] M. Hebert, and T. Kanade. Outdoor scene analysis using range data. In *IEEE International Conference on Robotics and Automation.* 1986.

[10] M. Hebert, C. Thorpe, S. Dunn, J. Cushieri, P. Rushfeldt, W. Girodet and P. Schweizer. A Feasibility Study for Long Range Autonomous Underwater Vehicle. In *Proceedings of Fifth International Symposium on Unmanned Untethered Submersible Technology.* June, 1987.

[11] M. Hebert and T. Kanade. 3-D vision for outdoor navigation by an autonomous vehicle. In *Proc. Image Understanding Workshop.* Cambridge, 1988.

[12] M. Hebert, T. Kanade, and I. Kweon. *3-D Vision Techniques for Autonomous Mobile Robots..* Technical Report CMU-RI-TR-88-12, Carnegie Mellon University, The Robotics Institute, August, 1988.

[13] M. Hebert, C. Caillas, E. Krotkov, I Kweon, and T. Kanade. Terrain Mapping for a Roving Planetary Explorer. In *IEEE Conference on Robotics and Automation.* 1989.

[14] M. Hebert. Building and Navigating Maps of Road Scenes Using an Active Sensor. In *IEEE Conference on Robotics and Automation.* 1989.

[15] T. Kanade, C. Thorpe, and W. Whittaker. Autonomous Land Vehicle Project at CMU. In *Proc. 1986 ACM Computer Conference.* Cincinnati, February, 1986.

[16] T. Kanade and C. Thorpe. *CMU Strategic Computing Vision Project Report: 1984 to 1985.* Technical Report, Carnegie Mellon University, The Robotics Institute, 1985.

[17] K. Kluge and C. Thorpe. Explicit Models For Road Following. In *IEEE Conference on Robotics and Automation.* 1989.

[18] B. Krogh and C. Thorpe. Integrated Path Planning and Dynamic Steering Control for Autonomous Vehicles. In *IEEE Conference on Robotics and Automation.* 1986.

[19] I. Kweon, M. Hebert, and T. Kanade. Perception for Rough Terrain Navigation. In *Proc. of SPIE on Mobile Robots.* SPIE, November, 1988.

[20] I. Kweon, M. Hebert and T. Kanade. Sensor fusion of range and reflectance data for outdoor scene analysis. In *Proc. Space Operations Automation and Robotics.* 1988.

[21] L. Matthies and S. Shafer. Error modeling in stereo navigation. *Journal of Robotics and Automation, Vol. 3* , 1987.

[22] L. Matthies, R. Szeliski and T. Kanade. Kalman filter-based algorithms for estimating depth from image sequences. In *Proc. Image Understanding Workshop.* Cambridge, 1988.

[23] H. Moravec and A. Elfes. High-resolution maps from wide angle sonar. In *Proc. IEEE International Conference on Robotics and Automation.* St. Louis, 1985.

[24] D. Reece and S. Shafer. An Overview of the PHAROS Traffic Simulator. In *2nd International Conference on Road Safety*. September, 1987.

[25] S. Shafer, A. Stentz, and C. Thorpe. An architecture for sensor fusion in a mobile robot. In *IEEE International Conference on Robotics and Automation*. 1986.

[26] J. Singh et al. *NavLab: An Autonomous Vehicle*. Technical Report, Carnegie Mellon Robotics Institute, 1986.

[27] A. Stentz and C. Thorpe. An Architecture for Autonomous Vehicle Navigation. In *Proceedings of the Fourth International Symposium on Unmanned Untethered Submersible Technology*. University of New Hampshire Marine Systems Engineering Lab, June, 1985.

[28] A. Stentz. . PhD thesis, Carnegie Mellon University School of Computer Science, 1989.

[29] C. Thorpe. *FIDO: Vision and Navigation for a Robot Rover*. PhD thesis, Carnegie Mellon University, 1984.

[30] C. Thorpe. Path Relaxation: Path Planning for a Mobile Robot. In *AAAI-84*. August, 1984.

[31] C. Thorpe, M. Hebert, T. Kanade and S. Shafer. Vision and Navigation for Carnegie Mellon Navlab. *Annual Review of Computer Science* 2, 1987.

[32] C. Thorpe and T. Kanade. *1987 Year End Report for Road Following at Carnegie Mellon*. Technical Report CMU-RI-TR-88-4, Carnegie Mellon University, The Robotics Institute, April, 1988.

[33] C. Thorpe, M. Hebert, T. Kanade and S. Shafer. Vision and navigation for the Carnegie-Mellon Navlab. *PAMI* 10(3), 1988.

[34] C. Thorpe and T. Kanade. *1986 Year End Report for Road Following at Carnegie Mellon*. Technical Report CMU-RI-TR-87-11, Carnegie Mellon University, The Robotics Institute, 1987.

[35] C. Thorpe and T. Kanade. *1987 Year End Report for Road Following at Carnegie Mellon*. Technical Report CMU-RI-TR-88-4, Carnegie Mellon University, The Robotics Institute, 1988.

[36] C. Thorpe and T. Kanade. *1988 Year End Report for Road Following at Carnegie Mellon*. Technical Report CMU-RI-TR-89-5, Carnegie Mellon University, The Robotics Institute, 1989.

[37] R. Wallace, A. Stentz, C. Thorpe, H. Moravec, W. Whittaker, and T. Kanade. First Results in Robot Road-Following. In *Proc. IJCAI-85*. August, 1985.

[38] R. Wallace, K. Matsuzaki, Y. Goto, J. Webb, J. Crisman, and T. Kanade. Progress in Robot Road Following. In *Proc. IEEE International Conference on Robotics and Automation*. April, 1986.

[39] W. Whittaker. *Terregator - Terrestrial Navigator*. Technical Report, Carnegie-Mellon Robotics Institute, 1984.

AN INTELLIGENT TACTICAL TARGET SCREENER

Gil J. Ettinger Tod S. Levitt

Advanced Decision Systems
1500 Plymouth Street
Mountain View, CA 94043

ABSTRACT

The problem of digital radar image interpretation compounds the conventional machine vision issues of representation, matching, search strategy, and hypothesis verification by greatly increasing the number of alternative explanations potentially considered. In processing SAR imagery for military force unit identification, the subject of this paper, high false alarm rates are obtained since the imagery is non-literal it incorporates reflections from many sources, often without providing enough cues to distinguish between these sources. As a result, multiple types of knowledge are required to provide context and help resolve conflict among alternative interpretations. The Intelligent Tactical Target Screener system (INTACTS) defines an approach that employs several types of knowledge in addition to imagery, such as military doctrinal knowledge, terrain information, other intelligence sources (other than imagery), and SAR phenomenology, in different ways that together comprise a unified and coherent approach to image exploitation. This approach, based on model-based Bayesian probabilistic inference, defines a hierarchical reasoning scheme that gradually focuses on the best force presence explanations while reducing the high combinatorics inherent in this class of problems.

1. INTRODUCTION

This paper describes the INTACTS (INtelligent TACtical Target Screener) system whose objectives are to provide a means for enhancing and automating digital radar imagery interpretation for applications including tactical intelligence missions, military situation assessment, and target recognition. In order to perform these tasks, the system takes as inputs a set of radar images, collection parameters, and contextual knowledge including terrain and military doctrinal information, and outputs its inferences about forces and their locations given the available evidence. The forces of interest are generally high level forces such as battalions and regiments.

INTACTS is a prototype system built under the ADRIES (Advanced Digital Radar Imagery Exploitation System) program. This program's goals are to develop a technology base for automating the exploitation of imagery, provide a testbed for technology development and evaluation, and baseline functional capabilities and transition them to the operational world [Program Plan - 86] [Levitt et al. - 88a].

INTACTS is founded on a theory of model-based Bayesian probabilistic inference. Models represent knowledge of the structural organization and formations of military units as well as terrain knowledge that can be used to provide evidence in support or denial of the presence of military forces at given locations. A probabilistic certainty calculus [Levitt et al. - 1986a] specifies how evidence extracted from synthetic aperture radar (SAR) imagery and terrain databases is matched against the models and combined to infer the

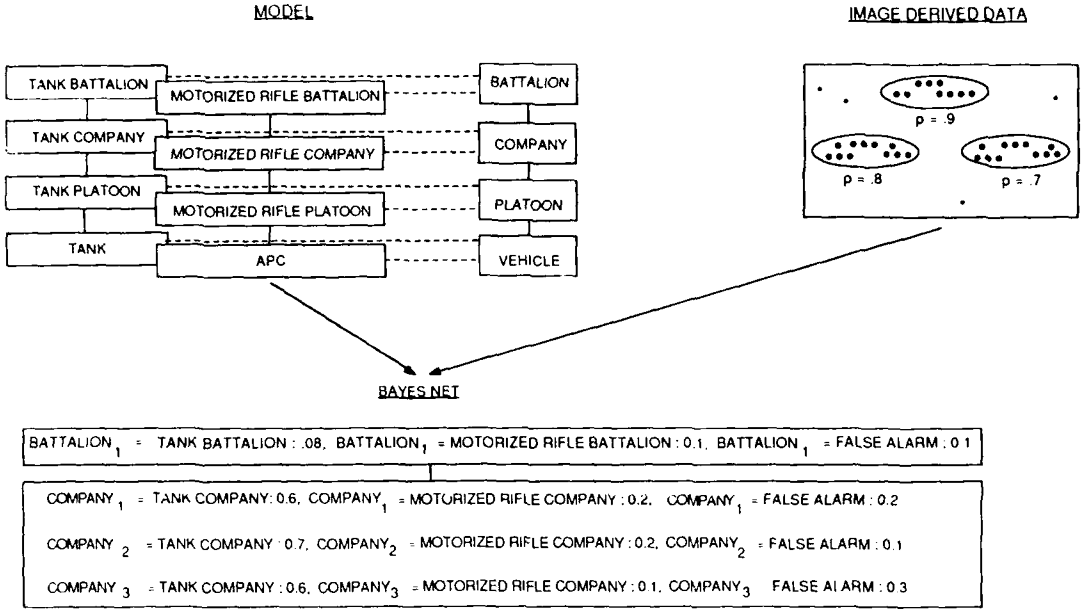

MODEL
IMAGE DERIVED DATA
TANK BATTALION
MOTORIZED RIFLE BATTALION
BATTALION
TANK COMPANY
MOTORIZED RIFLE COMPANY
COMPANY
TANK PLATOON
MOTORIZED RIFLE PLATOON
PLATOON
TANK
APC
VEHICLE
p = .9
p = .8
p = .7
BAYES NET
BATTALION1 = TANK BATTALION : .08, BATTALION1 = MOTORIZED RIFLE BATTALION : 0.1, BATTALION1 = FALSE ALARM : 0.1
COMPANY1 = TANK COMPANY: 0.6, COMPANY1 = MOTORIZED RIFLE COMPANY : 0.2, COMPANY1 = FALSE ALARM : 0.2
COMPANY2 = TANK COMPANY : 0.7, COMPANY2 = MOTORIZED RIFLE COMPANY : 0.2, COMPANY2 = FALSE ALARM : 0.1
COMPANY3 = TANK COMPANY : 0.6, COMPANY3 = MOTORIZED RIFLE COMPANY : 0.1, COMPANY3 FALSE ALARM : 0.3

Figure 1: Model-based Bayesian Inference Approach

presence or absence of military units. In particular, radar data, forces, terrain or other entities that have been modeled in the certainty calculus can be used as evidence for inferences output by the system.

The model database of INTACTS implicitly contains all possible chains of inference that the system can use to draw any conclusion. However, any information whatsoever can be used for control in the system without danger of circular reasoning in its conclusions. This distinction between control and inference is due to the requirement that any output hypotheses about forces in a region must be supported by image and terrain evidence as specified in the model database, thus resulting in a clean split between inference and control. For example, other source intelligence (other than imagery) can be used to direct search, make predictions, or act as a trigger to activate agents or other processing, but it cannot be fused to provide evidential support for system outputs unless it is modeled in the model database and in the certainty calculus.

Figure 1 shows the concept behind the model-based Bayesian inference in *INTACTS*. *Evidence, such* as clusters of detections, are matched against the geometry *of* formations that are explicitly represented in the model database. Matches lead to generation of alternative hypotheses of the presence of forces, such as batallions and companies, that are dynamically created and instantiated in a data structure called a Bayesian network. Additional evidence is accrued to refine the hypotheses in terms of their force and deployment types and to compute refined beliefs associated with the hypothesized forces. Under this paradigm the system attempts to interpret the scene by focusing on the correct hypotheses in a dynamic hierarchical reasoning approach.

The following section describes our model-based approach to image exploitation. Section 3 provides an overview of the INTACTS system architecture. *Section* 4 provides an example of a sample run, followed by concluding remarks.

2. MODEL-BASED BAYESIAN INFERENCING

Inference in INTACTS is performed over a space of hierarchically linked hypotheses. The hypotheses are generally of the form: *There is a military force of type F in deployment D at world location L at time T.* The hierarchy in the hypothesis space corresponds to the hierarchy inherent in military doctrine represented by the model database. Thus company hypotheses are linked to their component vehicle hypotheses in one direction and are grouped together to form battalion hypotheses in the other direction. The model database also specifies different classes for the variables making up the hypothesis definition. It thus defines means for refining company hypotheses to such refined classes as tank company and motorized rifle company hypotheses. Hierarchical Bayesian inference was introduced by Gettys and Willke [Gettys and Willke - 69], Schum and Ducharme [Schum - 80], Kelly and Barclay [Kelly and Barclay - 73] and has been carried forward by others, for example [Levitt et al. - 1986a], [Pearl - 86b], and [Binford - 87].

The inference process is performed in a hierarchical manner in order to achieve (1) favorable combinatorics and (2) a reasoning behavior of focusing on the best interpretation in a coarse to fine manner. The hierarchical structure of the hypothesis space is decomposed into two orthogonal axes. In one direction, the hypotheses are broken down by structure – regiments are composed of battalions, which in turn are composed of companies. In the other direction, the hypotheses are broken down by type – a company may be a tank or motorized rifle company, which in turn may be in a defensive, assembly, road movement, or road halt deployment.

The combinatoric benefit is two-fold [Ettinger - 88]. By explicitly using a structure, or part-of, hierarchy we refrain from ever considering a wide class of invalid configurations. The part-of hierarchy partitions the search space into potential feature matches belonging to the same sub-force. Many feature matches can be removed from consideration by virtue of belonging to a different sub-force than the one under current consideration. The overall possible search space consists of all possible assignments of image features to model features. While a recognition system would not normally explore all these configurations, it is preferable to start with as small a search set as possible. For a model of M features and an image of I features, the search space (for one-to-one feature matching, assuming $I > M$) for a non-hierarchical recognizer is: $\frac{I!}{(I-M)!}$. If the model is decomposed into N parts, the search space reduces to: $N\frac{I!}{(I-\frac{M}{N})!}$. This scheme achieves favorable results as long as the recognition of the sub-forces can easily be combined to reach an interpretation for the whole model. This check for consistency among the components can usually be done quickly since the number of sub-forces is small. For example, the search space is much smaller if we structure the search for a regiment as a set of battalions that in turn consist of company clusters rather than defining a regiment directly as configurations of companies. Given 3 battalions of 4 companies each in a regiment model, and 20 potential company clusters in the imagery, the search space of the non-hierarchical recognizer would be 6×10^{13} configurations, while the search space for the hierarchical recognizer would be 3×10^5.

The efficiency benefit of a type, or is-a, hierarchy is that all the features do not have to be considered at once. By initially matching only the coarse features, we avoid exploring the whole search space. The fine features are considered later to refine possibly inexact coarse configurations.

By virtue of this hierarchical nature, the system supports multiple inference paths. This flexibility is desirable in order to allow the inference process to proceed in the direction that exploits the most available evidence or the one that makes best use of the available resources. For example, if it is attempting to confirm or deny a tank regiment force, the system may commence by forming several hypotheses about locations of company-sized forces (based on clusters found in the images) and then aggregate them into battalion-sized hypotheses, and in turn into regiment-sized hypotheses, or at any point, it may attempt to refine the hypotheses by their force type (using spot images or terrain constraints) or deployment geometries (if they match a distinct deployment formation).

Although control processing in INTACTS can be complex, the structure of inference follows a pattern based on the models of military forces. These models consist of force structure and formation descriptions. The hypotheses are generated by hierarchical and partial matching of these military force models to evidence available in radar imagery. Evidence of the truth of a parent hypothesis is accrued numerically from probabilistic estimates about the sub-hypotheses that comprise the parent hypothesis as well as from other evidence sources, such as terrain, that directly support the parent hypothesis.

The actual reasoning over the hypotheses that takes place in the Bayes net is in the form of evaluating belief tradeoffs among *configurations* of hypotheses. The system is actually reasoning over sets of mutually exclusive consistent configurations of hypotheses. This mapping from individual hypotheses to configurations of hypotheses is necessary since the conflict between individual hypotheses may render them dependent on each other. Each node in the Bayes net groups together conflicting or dependent hypotheses, generates consistent configurations of the hypotheses, and evaluates the belief of each consistent configuration in a uniform manner using the laws of probability. An example of the generation of the Bayes net is shown in Figure 2. This figure shows the generation of two conflicting battalion hypotheses based on four company hypotheses. The diagram at the top shows formation region constraints of possible battalions fitted to company formation constraints. These hypotheses are grouped into nodes in the Bayes net where the system reasons over the consistent configurations shown. The belief of an individual hypothesis may then be calculated by adding the beliefs of the configurations that include the hypothesis of interest. The conditional probability matrix associating battalion configurations with company configurations is calculated from results of matching the companies to battalion formations.

One of the major motivations for selecting probability theory as the underlying technology for the numerical accrual of evidence is that Bayesian inference is a well-developed scientific theory and already exists for probabilistic evidential reasoning; see for example [D. von Winterfeldt and W. Edwards - 86]. This approach, though, requires us to lay out, a priori, the links between evidence and hypotheses in the models over which the system will reason. These links are the product of building probability models for the evidence sources.

A simplified picture of the inference process is summarized is Figure 3. The goal is to evaluate alternative hypotheses; but in order to accomplish that task we gather evidence from multiple evidence sources using predefined probability models, match the hypotheses to formation models to generate conditional probabilities relating the hypotheses, and map the hypotheses to mutually exclusive consistent configurations that we actually reason over.

2.1 MODEL DATABASE

The model database supports the hierarchical structure of the inference system by describing the forces according to their force/deployment types and structure. The models define the sub-part decomposition, feature abstraction, and matchable features of the force units.

Each force instance is a node on a graph. In one direction, part-of links point to the node's super-force

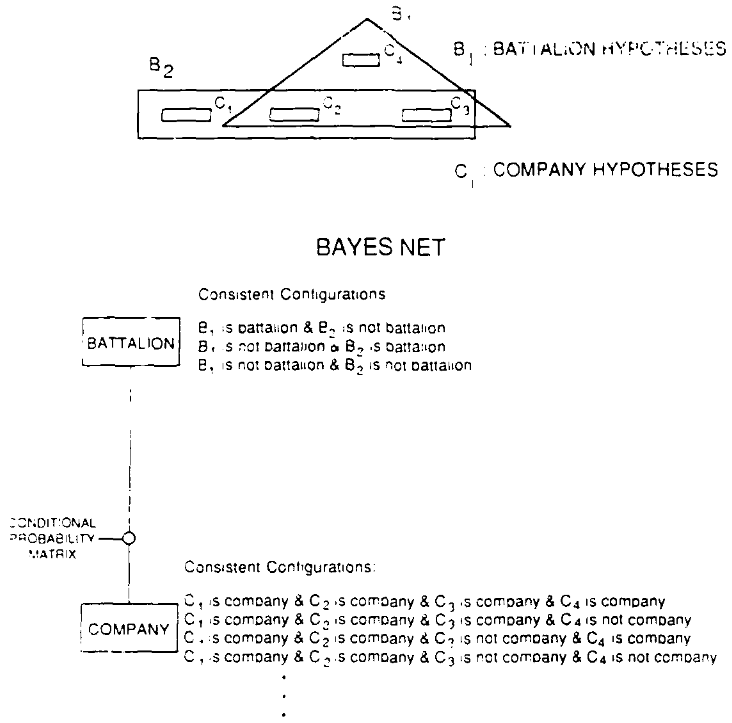

B1
B2
C1
C2
C3
C4
Bi : BATTALION HYPOTHESES
Ci : COMPANY HYPOTHESES
BAYES NET
Consistent Configurations
BATTALION
B1 is battalion & B2 is not battalion
B1 is not battalion & B2 is battalion
B1 is not battalion & B2 is not battalion
CONDITIONAL PROBABILITY MATRIX
Consistent Configurations:
COMPANY
C1 is company & C2 is company & C3 is company & C4 is company
C1 is company & C2 is company & C3 is company & C4 is not company
C1 is company & C2 is company & C3 is not company & C4 is company
C1 is company & C2 is company & C3 is not company & C4 is not company

Figure 2: Bayes Net Generation Example

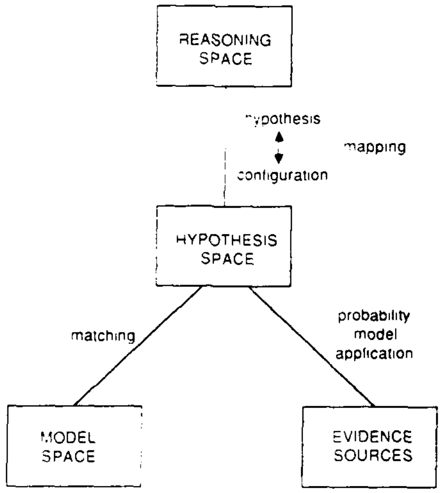


Figure 3: Simplified Schematic of Inference

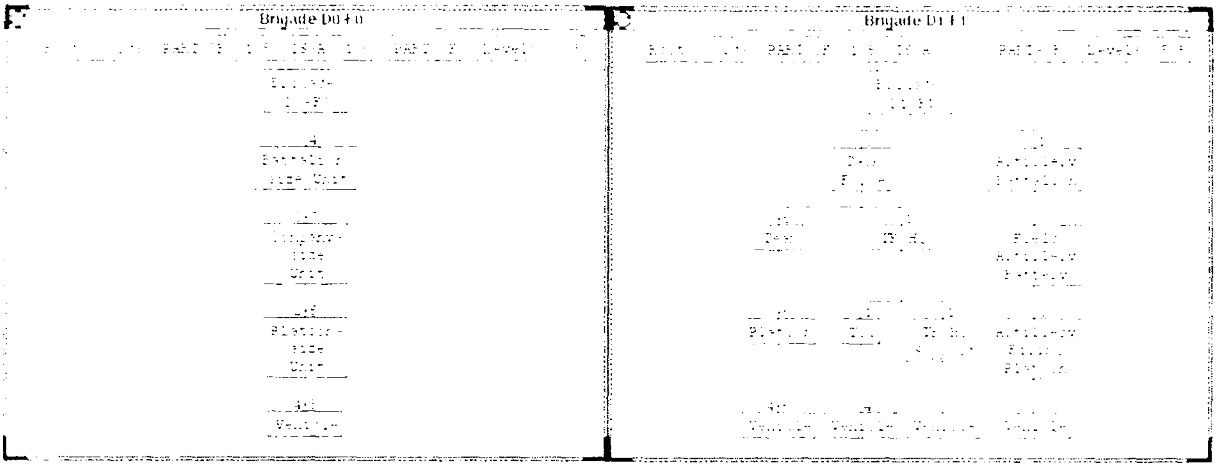


Figure 4: (A) Coarsest Level of Is-A Hierarchy (left), and (B) Force Type Refinement of Is-A Hierarchy (right)

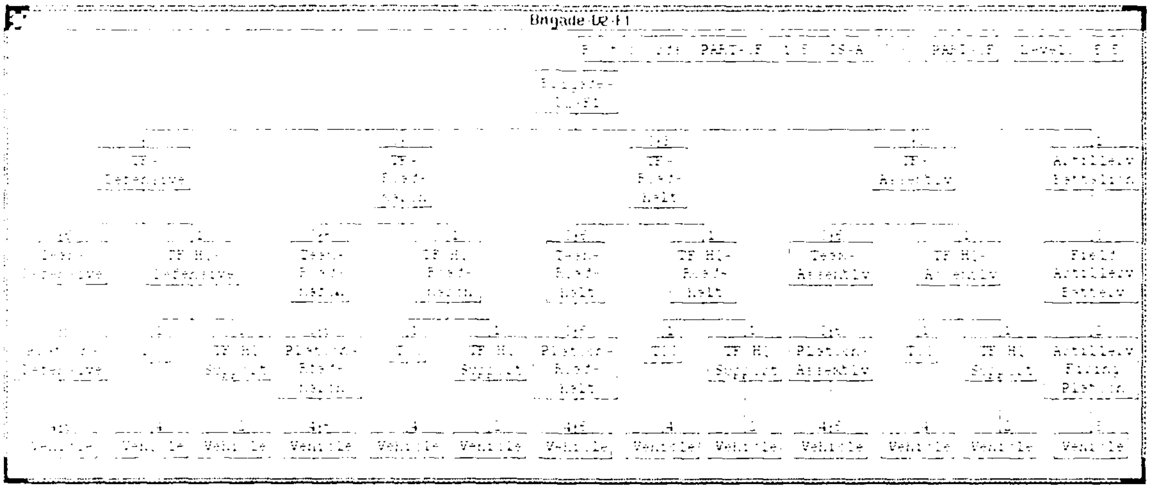
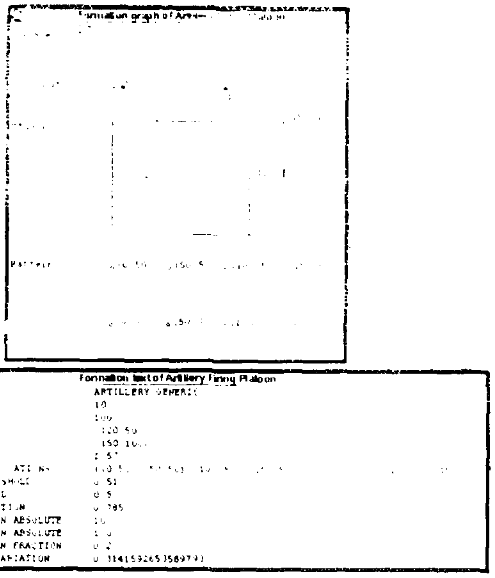


Figure 5: Deployment Type Refinement of Is-A Hierarchy



Formation text of Artillery Firing Platoon

```
DEPLOYMENT NAME                          ARTILLERY GENERIC
MIN SPACING                              10
MAX SPACING                              100
SMALLEST REGION                          (20 50)
LARGEST REGION                           (150 100)
REGION ORIENTATION                       1.57
NOMINAL MODEL POINT LOCATIONS            ((0 5[illegible]
MATCH EVALUATION THRESHOLD               0.51
MATCH NUMBER THRESHOLD                   0.5
MAX ORIENTATION VARIATION                0.785
MIN DISTANCE VARIATION ABSOLUTE          10
MAX DISTANCE VARIATION ABSOLUTE          100
MAX DISTANCE VARIATION FRACTION          0.2
MAX DIRECTION ANGLE VARIATION            0.3141592653589793
```

Figure 6: Sample Formation Description

structure (e.g. artillery battery is part of artillery battalion) and in the orthogonal direction is-a links point to the node's generalized type (e.g. artillery battalion is a type of battalion). Attached to each node are descriptions of its possible formations. A section of a sample model database is shown in Figures 4 through 6. These figures show 2 successive refinements of the coarsest level model description. This coarsest part-of hierarchy, Figure 4A, has abstracted features common to all units of approximately the same size and describes the models in generic terms. Thus a brigade consists of four battalions, each consisting of three to seven companies. Figure 4B shows a force type refinement of the part-of hierarchy and Figure 5 further refines the models by deployment type. The force and deployment type refinements are actually interleaved down further is-a refinements so at each description level the system can choose either direction for further refinement. The links shown in these figures are part-of links. Is-a links are not shown, but are present across the figures. For example, both *Task Force* and *Artillery Battalion* in Figure 4B have is-a links to *Battalion-size unit* in Figure 4A. Figure 6 depicts a formation description for the artillery firing platoon. Similar descriptions are attached to every model node.

2.2 EVIDENCE SOURCES

Evidence that is to be fused directly into the inference process must be modeled in the system so that features extracted from the evidence source can be used to generate probabilities in support or denial of the generated hypotheses. The probability space then provides a uniform mechanism of combining evidence from different sources. It follows that to extend INTACTS to a full fusion system, it will be necessary to do the research on the knowledge representation of the additional sources of information, and the probabilistic relationships between these sources and the entities already accounted for in models and the calculus.

Currently, probability models have been generated for radar derived data (detections and clusters) and local terrain data (low-level terrain attribute suitability for forces) [Levitt et al. - 1986a] [Levitt et al. - 1986b] [Levitt et al. - 1987]. Detection and cluster models allow calculation of probabilistic support for *force unit vs* *m. force unit* hypotheses based on features extractable from imagery. Work is also proceeding on providing probabilities for different vehicle type hypotheses from observed image features. Local terrain models support calculation of evidence for different force types based on direct underlying terrain characteristics such as slope, vegetation, etc. These models have been built through performance-based knowledge elicitation, statistical analysis of available data, and knowledge engineering sessions with domain experts. Efforts to extend these probability models for additional force types and evidential features are continuing.

Knowledge sources for which we do not have a probability model can be incorporated in other ways. One option is to use the information for control rather than inference. Since the control and inference processes are separable we can use such knowledge as signal intelligence for which we do not have probability models to prioritize the search regions or direct hypothesis generation paths.

A method of incorporating contextual constraint knowledge is to adjust the formation models to reflect situational characteristics. The formation models in the system are nominal or average models that do not take into account particular prevailing conditions. In order to get more accurate matches of the model to the image features, the model should be refined to reflect such rules as *all firing units must deploy on the same elevation, area of deployment must have ingress and egress*, or *forces cannot deploy in no-go areas.* Many of these rules do not currently have associated probability models since they perform complex relations of terrain attributes and the current military situation. The constraining power of these rules, though, is used to refine models that have been coarsely matched to image features and therefore reflects in the hypotheses' beliefs through reevaluation of the match. This refinement action, termed situation model adjustment, refines the hypotheses by incorporating constraints inherent in the prevailing situation as defined by the interaction of terrain and military doctrine.

2.3 HYPOTHESIS GENERATION

Hypotheses are generated from the results of matching sub-units into parent force units. All matches falling within the constraints of the models may potentially be generated even though only the best ones are pursued first. The evaluation of the match is then used as evidence by defining the conditional probabilities associating the sub-units with the parent unit. Initial hypothesis generation is performed by using the results of clustering or detection. An important distinction between clustering and matching is that clustering does not generate geographically overlapping hypotheses while matching potentially does. This decision is made to greatly improve the combinatorics at the lower force levels. Clustering is thus intended to generate coarse lower level hypotheses, that may later be refined. Since we are ultimately interested in the location and identity of the higher-level force hypotheses, and the false alarm rate is very high at the lower levels, we use clustering at these lower levels to greatly improve the system's efficiency. Clustering is also generally useful in this domain of military force recognition since the inter-company spacing in battalions is generally larger than the inter-vehicle spacing in companies, thus making it feasible to use clustering to segment the detection set into separable groups. Such characteristics are usually necessary for effective segmentation when using coarse constraints.

All matches, including clusters, can be refined by incorporating additional features in the match. The different matchers used to enable this form of hierarchical matching are:

Global parameter matcher: inter-sub-unit spacing and number of sub-units (features used by clustering).

Region matcher: incorporation of region shape and area with above global parameters.

Spatial pattern matcher: geometric properties of pairs and triples of sub unit locations.

The matchers are used to both generate and evaluate hypotheses at all levels of the part-of and is-a hierarchy.

2.4 REASONING SPACE

Reasoning in INTACTS is not performed directly on hypotheses in order to account for hypothesis interdependencies induced by different types of conflict. The simplest form of conflict, and one always present, is existence: a region does contain a military force or does not contain that force. Additional conflict sets may be due to location, for geographically overlapping hypotheses, or type, for different classes of hypotheses such as tank battalion or motorized rifle battalion. As a result, INTACTS reasons over the complete mutually exclusive sets of consistent configurations of the generated hypotheses as shown in Figure 2. These configurations can be viewed as generated from matching data and pruned by inter-hypothesis conflicts. In practice a tradeoff exists between using few matching features to generate fewer configurations while being less descriptive and using many features to generate more configurations while being more descriptive. The latter approach also benefits from being able to prune many of the configurations due to the presence of more potential conflict. It is a much more time-consuming process, though, which is why a hierarchical approach is adopted where by starting out with a few general features, and progressively incorporating more detailed features for the best hypotheses, many of the possible configurations are pruned by the time the fine features are used.

The evidential accrual for the hypotheses' configurations is performed by using a Bayesian network since this paradigm supports the propagation of evidential probabilities through a hierarchical tree of hypotheses in a coherent, stable, and efficient manner. Pearl's algorithm is used for belief propagation since that technique can be implemented as an active network of identical autonomous processors in a distributed environment [Pearl - 86a]. Therefore, when propagating belief through the network, equilibrium is guaranteed to be reached in time proportional to the network diameter. Pearl specified his propagation algorithm for static hypothesis spaces. His results have been extended for dynamic hypothesis generation in order to apply to the INTACTS application. New high level hypotheses are generated from aggregation of lower level hypotheses, while refinement stages generate more possibilities for each individual hypothesis. Conflicting or related hypotheses are then merged into a single Bayes node, which defines a belief vector indicating the current belief that each possible configuration of the hypotheses is true. Bayes evidence nodes are generated for the different evidence sources in the system and are attached to the Bayes nodes that contain the hypotheses for which the evidence was accrued. Beliefs are then propagated across the different levels of the Bayes net via conditional probability matrices that relate parent to child hypotheses via formation and terrain constraints. By using this reasoning scheme, we can associate a deductive chain of evidence with any final hypothesis' belief even though the actual processing may have been complex, with numerous feedback loops and multiple levels of resolution.

2.5 CONTROLLING THE INFERENCE

INTACTS employs a control methodology based on applying utility theory to model-based Bayesian inference [Levitt et al. - 88b]. Candidate processing actions are selected by utility computations based on the estimated value and cost of each action. Executable actions are then selected by maximizing total value bounded by a specified total cost constraint. Value is defined in terms of a metric measuring distance from the goal as specified in the model database and the current belief of the hypotheses to be acted on. Action

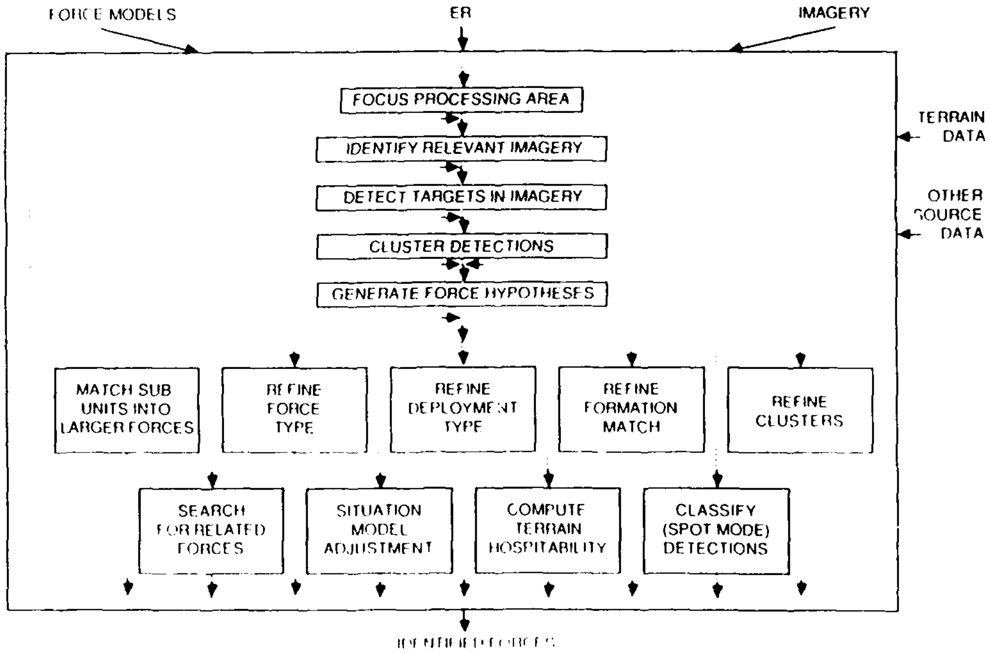

FORCE MODELS
ER
IMAGERY
TERRAIN DATA
OTHER SOURCE DATA
FOCUS PROCESSING AREA
IDENTIFY RELEVANT IMAGERY
DETECT TARGETS IN IMAGERY
CLUSTER DETECTIONS
GENERATE FORCE HYPOTHESES
MATCH SUB UNITS INTO LARGER FORCES
REFINE FORCE TYPE
REFINE DEPLOYMENT TYPE
REFINE FORMATION MATCH
REFINE CLUSTERS
SEARCH FOR RELATED FORCES
SITUATION MODEL ADJUSTMENT
COMPUTE TERRAIN HOSPITABILITY
CLASSIFY (SPOT MODE) DETECTIONS
IDENTIFIED FORCES

Figure 7: INTACTS Control Flow

value is increased with higher belief for the source hypotheses or with decreased distance to the goal. Cost is defined as the average processing time for the action.

The possible processing actions in INTACTS are shown in Figure 7, where the decisions to select certain control flows are made by utility computations based on the current state. This control strategy supports flexible reasoning since the inference methodology does not place tight constraints on the action sequence or order of evidential accrual, and the model database supports multiple reasoning paths across different combinations of part-of and is-a links.

3. SYSTEM ARCHITECTURE

The INTACTS system is built as a distributed set of software agents communicating by message passing. In the current prototype system being developed, agents are hosted on Sun workstations, Symbolics LISP machines, and a Warp systolic array processor, with additional implementations to be developed on a Connection Machine and an Encore multiprocessor. The agent decomposition for INTACTS is shown in Figure 8. Each agent is a functional module running as an independent virtual process and has a well-defined I/O specification. All agents have access, either directly or indirectly, to all databases. Agents may also have internal databases for monitoring their current state. Message passing is handled by a uniform communication system allowing dynamic logical address specification independent of the identity of physical host.

In the current system, functional planning and resource allocation are loosely coupled via the Agenda. The Control agent plans system processing steps in order to fulfill a user specified goal of identifying one or

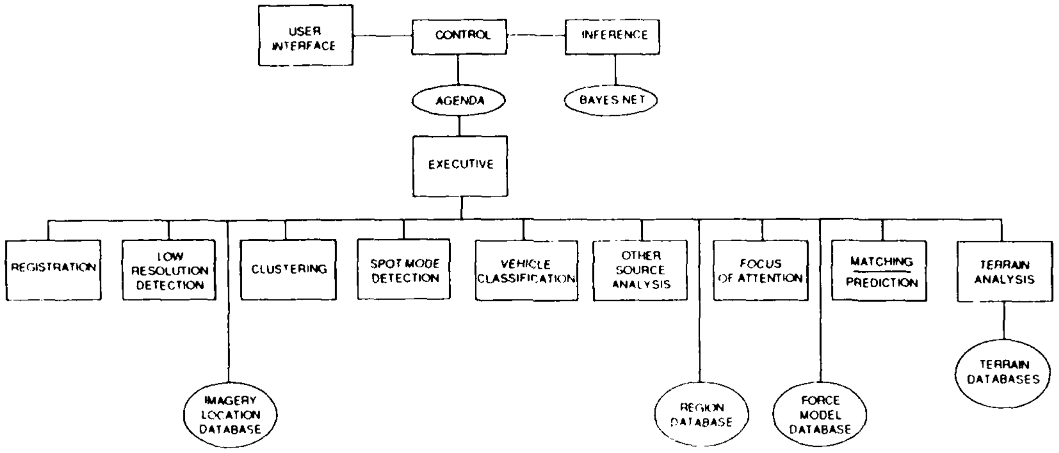


Figure 8: INTACTS System Architecture

more force types in given regions. In INTACTS, goals are usually entered via an exploitation requirement that directs the system to confirm or deny the presence of a particular type of force in a given region. Control posts its processing actions on the Agenda which maintains their priority and forwards them to the Exec agent for distribution to the appropriate functional agent. The Exec [Turner et al. - 88] monitors system status and performs resource allocation by dynamically routing I/O channels and remotely starting up new agents to alleviate bottlenecks.

In planning processing actions, Control closely interacts with Inference to examine the system state in the resident Bayes net. Inference maintains the Bayes net by grouping hypotheses into consistent configurations, propagating evidence throughout the net, and calculating beliefs of individual hypotheses.

Most of the functional agents in INTACTS provide evidential or control information that is used by Inference and Control. The Focus agent produces a prioritized list of regions to search based on terrain and available other source intelligence. Imagery covering these regions is processed by the detection agents (multiple agents for different resolutions) to produce candidate vehicle detections. The imagery is registered to the resident terrain map by the Registration agent so that all reasoning is done in world coordinates. *Detections and clustered detections*, as processed by the Clustering agent, are used to initialize hypotheses in the Bayes net. Higher level and refined hypotheses are generated and evaluated by the Matching agent. Any unmatched sub-units or regions are then used as predicted areas for verification and further processing.

Additional sources of evidence are provided by the Vehicle Classification and Terrain Analysis agents. Vehicle Classification calculates probabilities *over* the set of vehicle types for high resolution detections. Terrain Analysis calculates terrain hospitability probabilities for force hypotheses based on local terrain attributes. The Terrain agent also supports the processing of Focus as well as formation refinement for situation model adjustment.

	REGION AREA (sq km)	VEHICLES		COMPANIES		BATTALIONS	
GROUND TRUTH	94	65		6		2	
		GROUND TRUTH FOUND	FALSE ALARMS GENERATED	GROUND TRUTH FOUND	FALSE ALARMS GENERATED	GROUND TRUTH FOUND	FALSE ALARMS GENERATED
AVAILABLE IMAGERY	76						
FOCUSING ATTENTION (TERRAIN ANALYSIS) (OTHER SOURCE CUES)	18						
DETECTION		54	2133				
CLUSTERING DETECTIONS		40	619	6	94		
FORCE HYPOTHESIS GENERATION		31	314	4	9		
HIGHER LEVEL FORCE MATCHING		31	237	4	7	2	3
EVIDENCE REFINEMENT (REFINE FORCE TYPE, FORMATION MODELS FORMATION MATCH, CLUSTERS)		31	224	4	6	2	2
TERRAIN EVIDENCE		31	183	4	5	2	0

Figure 9: Sample INTACTS Processing

4. SAMPLE PROCESSING RUN

In order to gain a perspective on the behavior of the INTACTS system we can monitor the system's performance during a test run of processing SAR imagery and compare its results to ground truth. Figure 9 shows these results for a sample run. In this scenario, two battalions were deployed in the region of interest. The battalions were composed of six total companies that contained 65 total vehicles. The performance results show a count of true and false hypotheses (in terms of region specification) at each level under consideration by INTACTS after execution of the actions listed in the leftmost column. This run demonstrates how the system focuses on its final scene interpretation through applications of evidential accrual, refinement, and matching actions. As a final answer it identified the two battalion regions even though the lower level features still included false alarms. Thus, INTACTS was able to successfully infer the higher level force units even though it did not examine the whole image, nor did it need to resolve all the conflict at the lower level units.

In addition, this run demonstrates how the system effectively reduces the false alarm rate during processing. As we apply the various actions, the false alarm rate for hypotheses currently under consideration decreases for each unit size. For example, the false alarm rate for vehicles decreased from 39.5 : 1 to 5.9 : 1 during processing. Furthermore, the false alarm rate decreased as we looked at higher level hypotheses. For example, after evidence refinement, the false alarm rate for vehicles was 7.2 : 1, for companies was 1.5 : 1, and for battalions was 1.0 : 1. Both of these false alarm reduction patterns are desirable for a screening system such as INTACTS.

Additional tests are being run on the system in order to measure its performance and limitations under different input and operating conditions.

5. CONCLUSION

The INTACTS system, developed under the ADRIES project, is an intelligent image exploitation system that integrates contextual and terrain reasoning to identify complex military forces in the viewed imagery. It defines means of using and combining knowledge sources supporting image exploitation in a uniform and coherent paradigm that reduces combinatorics and achieves effective hierarchical reasoning. Technical contributions of this work include:

- Development of a hypothesis modeling methodology based on:

    a probabilistic approach that supports:

        consistent belief values

        sound computational framework

        an environment for evidential fusion

    a hierarchical model representation that decomposes the system's knowledge by structure (sub-unit decomposition) and by type (force and deployment)

    an inference mechanism that reasons over the model representation in a hierarchical manner to achieve favorable combinatorics

    dynamic hypothesis generation across *multiple reasoning paths*

- Application of terrain and other contextual information to support the system's performance in three different ways:

    prioritize the regions where the system should focus its attention

    refine the models to reflect the prevailing situation

    provide evidence for directly supporting/refuting hypotheses

- Separation of the control and inference processes in order to achieve flexible and consistent reasoning

- Development of utility theory for use in the control process to plan the inference actions

- Development of a hierarchical matcher to match modei force formations to hypothesized forces using *varying degrees of detail in the geometric description*

- Development of a *distributed* system testbed consisting of:

    a uniform communication system interfacing the functional agents independently of the processes and machines on which they are run

    an executive agent to oversee the communication activity and dynamically allocate resources among the running agents.

At the current state of system development, INTACTS successfully infers the presence and locations of military units through the regimental level in high false alarm data, demonstrating the feasibility of the approach. Knowledge is being added to the system to allow it to perform more complex reasoning in a wide range of situations. Capability for software simulation of input data is also being developed to make statistically significant testing of INTACTS possible. The functional performance of INTACTS is being improved as this additional knowledge is being added while concurrent efforts are being made to increase the system's efficiency.

ACKNOWLEDGMENTS

Research and development of the INTACTS system has been supported under the ADRIES contract (U.S. Government Contract DACA76-86-C-0010) by the Defense Advanced Research Projects Agency (DARPA), the U.S. Army Engineer Topographic Laboratories (ETL), the U.S. Army Space Program Office (ASPO), and the U.S. Air Force Wright Avionics Laboratory (AFWAL). ADRIES is the joint work of many researchers at Advanced Decision Systems (ADS) in Mountain View, CA, Science Applications International Corporation (SAIC) in Tucson, AZ, and MRJ Inc. in Oakton, VA.

References

Binford - 87 T. Binford, and T. Levitt, "Bayesian Inference in Model-Based Machine Vision," in *Proceedings of AAAI Uncertainty in Artificial Intelligence Workshop*, Seattle, WA, July, 1987.

Ettinger - 88 G. Ettinger, "Hierarchical Object Recognition Using Libraries of Parameterized Model Sub-Parts," *Proceedings of IEEE Computer Vision and Pattern Recognition Conference*, Ann Arbor, Michigan, June, 1988.

Gettys and Willke - 69 C. Gettys and T. Willke, "The Application of Bayes' Theorem When the True Data State is Uncertain," *Organizational Behavior and Human Performance*, pp. 125-141, 1969.

Kelly and Barclay - 73 C. Kelly, III and S. Barclay, "A General Bayesian Model for Hierarchical Inference," *Organizational Behavior and Human Performance*, vol. 10, 1973.

Levitt et al. - 1986a T. Levitt, W. Edwards, B. Kirby, L. Winter, D. Morgan, S. Mori, S. Simmes, and F. Smith, "Design of a Probabilistic Certainty Calculus for ADRIES," Contract DACA76-86-C-0010, Advanced Decision Systems, Mountain View, CA, April, 1986.

Levitt et al. - 1986b T. Levitt, W. Edwards, and L. Winter, "Elicitation of A Priori Terrain Knowledge for ADRIES," Contract DACA76-86-C-0010, Advanced Decision Systems, Mountain View, CA, November, 1986.

Levitt et al. - 1987 T. Levitt, L. Winter, T. Eppel, T. Irons, and C. Neven, "Terrain Knowledge Elicitation for ADRIES: Part II," Contract DACA76-86-C-0010, Advanced Decision Systems, Mountain View, CA, October, 1987

Levitt et al. - 88a T. Levitt, G. Ettinger, T. Esselman, M. Black, T. Shaffer, R. Chestek, K. Riley, R. Drazovich, R. Kirby, "ADRIES Prototype System Development Program Annual Technical Report," Contract DACA76-86-C-0010, Advanced Decision Systems, TR-1131-02-1, May, 1988.

[Levitt et al. - 88b] T. Levitt, T. Binford, G. Ettinger, and P. Gelband, "Utility-Based Control for Computer Vision," in *Proceedings of AAAI Uncertainty in Artificial Intelligence Workshop*, Minneapolis, MN, August, 1988.

[Morgan - 89] D. Morgan, T. Miltonberger, G. Orr, "A Sensor Algorithm Expert System," in *Proceedings of SPIE/SPSE Symposium on Electronic Imaging: Advanced Devices and Systems*, Los Angeles, CA, January, 1989.

[Pearl - 86a] J. Pearl, "On Evidential Reasoning in a Hierarchy of Hypotheses," *Artificial Intelligence*, vol. 28, no. 1, pp. 9-15, February, 1986.

[Pearl - 86b] J. Pearl, "Fusion, Propagation, and Structuring in Belief Networks," *Artificial Intelligence*, vol. 29, no. 3, pp. 241-288, September, 1986.

[Program Plan - 86] DARPA, ETL, ADS, SAIC, MRJ, and TASC, "Advanced Digital Radar Imagery Exploitation System (ADRIES) Program Plan," Advanced Decision Systems, Mountain View, CA, October, 1986.

[Schum - 80] D. Schum, "Current Developments in Research on Cascaded Inference," in *Cognitive Processes in Decision and Choice Behavior*, T. Wallsten, Ed., Lawrence Erlbaum Press, Hillsdale, NJ, 1980.

[Turner et al. - 88] C. Turner, C. Winter, S. Sayre, "Parallel and Distributed Computing on the Sensor National Testbed," Science Applications International Corporation, Tucson, AZ, 1988.

[D. von Winterfeldt and W. Edwards - 86] D. von winterfeldt and W. Edwards, *Decision Analysis and Behavioral Research*, Cambridge University Press, Cambridge, MA, 1986.

OVERVIEW OF THE SCORPIUS PROGRAM

J. F. Bogdanowicz
Arthur Newman
Hughes Aircraft Company, Data Systems Division
El Segundo, California 90245

ABSTRACT

This paper presents an overview and current status of the SCORPIUS program. The SCORPIUS program is an applied research effort whose goal is to combine technologies from DARPA's Image Understanding and Computer architecture research areas in a real world application, automated exploitation of aerial imagery. The vision system under development as well as the parallel processing testbed which is being used to host the vision system is discussed.

INTRODUCTION

In July 1985, The Strategic Computing Object directed Reconnaissance Parallel processing Image Understanding System (SCORPIUS) program [SCORPIUS,1987; Bogdanowicz, 1987] was initiated. SCORPIUS is an applied research effort whose goal is to combine technologies from DARPA's Image Understanding and Computer architecture research areas in a real world application, automated exploitation of aerial imagery. Image exploitation is the process of extracting intelligence from image data. The quantity and quality of the products from various sensor systems are outpacing the improvements in the exploitation systems. The SCORPIUS program is focussed on correcting this imbalance; it is the first significant effort to implement a fully automatic image understanding system. The prototype system being developed will be evaluated over a large image testset. The introduction of this technology will be evolutionary instead of revolutionary, due to the difficulty of the task and the need for high confidence in the results produced by the machine. The goal of the program is to demonstrate object-directed image exploitation by a machine over a wide range of imaging conditions with performance equivalent to a novice image analyst in quality and within an order of magnitude in speed using scalable architectures.

The vision system under development is concerned with object-directed analysis. Object-directed analysis involves detecting, identifying, and counting modelled, man-made objects in known target locations and issuing a simplified assessment of the situation.

The SCORPIUS program has been structured around a risk reduction prototyping approach based loosely on the spiral development methodology [Boehm, 1988]. The major SCORPIUS prototype demonstrations have been used to focus on three key system requirements and include 1) automatic end to end operation of the vision system over two different scenarios with each scenario having two different target locations, 2) vision system support for a wide range of imaging conditions and complexities, and 3) vision system processing time within an order of magnitude of that required by an image analyst. Specific prototypes have focused on key technical issues related to the vision system and the parallel processing system environment developments. For each prototype, specific capabilities needed for the final system are incrementally addressed. Based on the results of a prototype demonstration, an assessment is made of how well the requirements were handled and refinements to the requirements and the plans for succeeding prototypes are then made. This approach has allowed the system to evolve in a flexible manner.

The second section describes the current structure of the vision system. The third section describes the application level of the vision system detailing the various discrimination levels implemented using the tools discussed in previous section. The fourth section provides, as an example, some of the technical details of the region and object discrimination levels of the system. The fifth section provides an overview of the parallel processing testbed that has been developed. Finally, in the sixth section, a summary of the major prototypes and milestones of the program are presented.

VISION SYSTEM DEVELOPMENT TOOLS AND STRUCTURE

The SCORPIUS program utilizes a model and knowledge based approach to computer vision [Berlin, Bogdanowicz and Diamond, 1987] with a toplevel conceptual structure as shown in Figure 1. The SCORPIUS vision system is actually composed of three major subsystems, the symbolic subsystem, the numeric subsystem, and an interface subsystem that connects the two. These subsystems reflect the underlying hardware architecture in addition to being conceptually distinct from a programming viewpoint. The specific hardware configurations on which these subsystems exist are intended to evolve throughout the life of the SCORPIUS project.

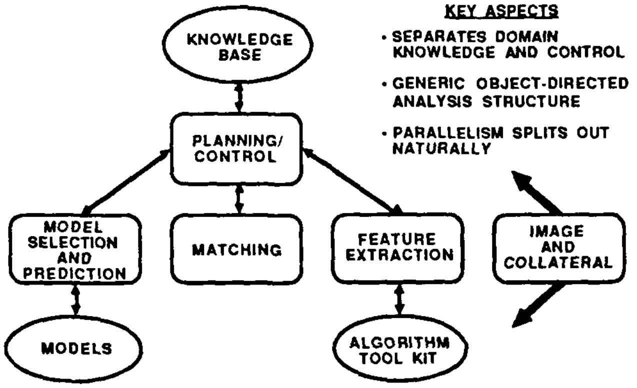

KNOWLEDGE BASE
PLANNING/ CONTROL
MODEL SELECTION AND PREDICTION
MATCHING
FEATURE EXTRACTION
IMAGE AND COLLATERAL
MODELS
ALGORITHM TOOL KIT
KEY ASPECTS
• SEPARATES DOMAIN KNOWLEDGE AND CONTROL
• GENERIC OBJECT-DIRECTED ANALYSIS STRUCTURE
• PARALLELISM SPLITS OUT NATURALLY

Figure 1. Toplevel Conceptual Organization of Vision System

Computer vision algorithms can be categorized as image-to-image, image-to-symbol, or symbol-to-symbol. Image-to-image algorithms typically transform 2-D arrays of pixels into other 2-D arrays of pixels. Image-to-symbol algorithms transform 2-D arrays into 1-D property lists, or feature vectors. Symbol-to-symbol algorithms often build abstract data structures out of property lists. This class of algorithms also includes manipulating pre-stored model information in various ways and in matching model information against data structures ultimately derived from pixels.

The symbolic subsystem runs symbol-to-symbol algorithms. It also has a planning / control function and acts as the top level executive for the vision system. This software is called the vision executive or the VX shell. The symbolic subsystem decides what algorithms to run and what data it needs. It does this by issuing two types of commands to the interface: algorithm execution requests, known as AXRs, and data extraction requests, known as DXRs. AXRs are interpreted by the interface which schedules and monitors their execution on a machine in the numeric subsystem. DXRs cause data to be transferred from somewhere within the numeric subsystem to the symbolic subsystem through the interface. The numeric subsystem runs image-to-image and image-to-symbol algorithms.

The planning component of the VX shell determines flow of control at the macro level. It decides what to do next. The planning component does not itself call any symbol-to-symbol algorithms nor does it create or receive AXRs or DXRs. These actions take place within a set of seven control expert systems. The control experts decide how something should be done. They do this by generating an appropriate sequence of AXRs and DXRs. Communication between the planner and the control experts takes place through an active data structure known as the interpretation graph. The interpretation graph maintains the processing state of the vision system at any point in time and also represents what has been identified in the image. Before continuing with a more detailed description of the VX shell, it will be helpful to discuss the different software layers that are used to build the symbolic subsystem.

In addition to the operating systems of the underlying machines, the symbolic subsystem software can be divided into four layers as shown in figure 2. The innermost layer is Common Lisp. The next layer is MOBIUS, a Hughes developed artificial intelligence tool. MOBIUS contains software for object oriented programming and for building rule based systems. The next layer is the vision execution (VX) shell. The VX shell is built out of various MOBIUS software objects. The outer layer is known as user application software.

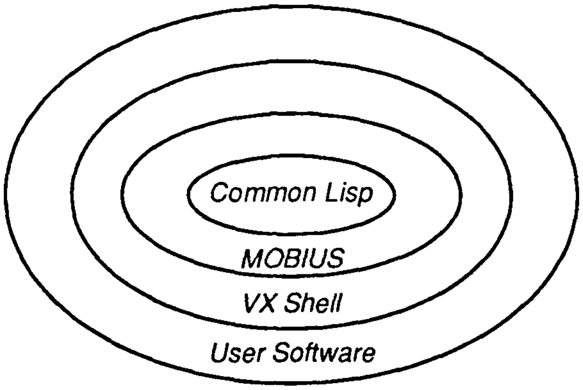


Figure 2. Symbolic subsystem software layers

MOBIUS has a frame system, a message passing system, a forward chainer, and a backward chainer. Rules and facts can be organized into inference objects (blackboards), as shown in figure 3. Inference objects can communicate with each other via message passing. The backward chainer is a syntactic variant of PROLOG. The forward chainer can be used to set up state space search problems. Rule conditions in the forward chainer can manipulate the STM of an inference object in a SIMD fashion. Both the backward and forward chainers use a similar syntax for rule conditions and have a wide variety of built-in predicates. Several mechanisms are provided for interfacing with Common Lisp.

The VX shell is constructed from a set of objects. As used here, the term object means a frame and zero or more associated methods (procedural Lisp code). A number of these objects are always present when the vision system is running. It is important to note that the VX shell software itself makes no commitment as to the representation of features or models. It contains no symbol-to-symbol algorithms. It is domain independent. It has built into it a set of assumptions underlying a particular philosophy as to how computer vision systems should be built. It also makes no hardware specific assumptions about the numeric subsystem. The VX shell also provides a small set of standardized frame patterns that can be instantiated and manipulated by the user software. These frames are used to build the interpretation graph.

One of the pre-defined VX shell objects is known as a plan. A plan is made up of a sequence of plan steps. Plan steps have an identical syntax to MOBIUS forward chaining rule conditions. Unlike forward chaining rules however, plans do not have pre-conditions or post-conditions. Plans are written ahead of time prior to a run of the vision system. Hence, plans are equivalent to scripts.

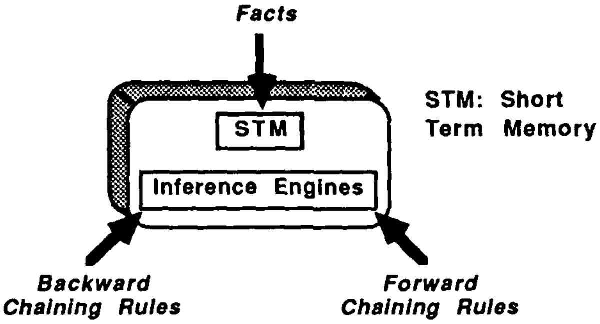


Figure 3. MOBIUS Inference Object

User software includes objects, sets of forward chaining rules, sets of backward chaining rules, plans, and Lisp functions. This software is for the most part domain dependent. It is used for building particular model representations and it manipulates anticipated representations of features obtained from the numeric subsystem.

The purpose of the VX shell software is to help support the primary SCORPIUS project goals of generality, image complexity, and speed. The VX shell software helps to meet the generality requirement by providing software that is scenario independent. Images that the vision system are required to process are of sufficient complexity that a large degree of ambiguity is present. It is believed that flexible planning / control capabilities are required in order to successfully disambiguate competing hypothesis that are inherent in intensity based imagery. The speed requirement is being met by utilitizing parallel processing hardware provided by DARPA. By suitable tailoring of the search strategy used in the planner, the VX software can provide a macro level flow that matches the space and time resources of the parallel processing hardware.

In figure 4 are shown the VX shell objects that can be manipulated by the user software. There are also VX shell objects not shown that are not manipulated by the user software but are instead used for internal message passing and bookkeeping. Two major classes of objects are shown, objects that make up the interpretation graph, or igraph, and the seven control experts. Two miscellaneous objects are also shown, the top level vision-system object and resource manager 2 (RM2) which communicates with the interface subsystem. The seven control experts manipulate various properties of the data and include features (FX), models (MHX, MPX, MAX), databases (HINT), control (PHX), and interpretation (MATCHER).

OVERVIEW OF THE VISION SYSTEM DISCRIMINATION LEVELS

Using the basic tools described in previous section, the vision system is implemented as a multi-level discrimination process as shown in Figure 5. The image level discrimination function consists of two subfunctions: cloud detection, and scene registration. Cloud detection is the process by which regions of the image that are obscured by clouds are identified and labelled. The scene registration process labels fixed structures and semantically important locations within the scene. Such information can be used to both guide the feature extraction and reasoning processes in other discrimination levels.The cloud detection process classifies the pixels of the original image as being "cloud" or "not cloud". The result of this classification process is actually a probabilistic map of the entire image, with each pixel being assigned a probability of being a cloud. Pixels in dense cloud areas have a high probability of being a cloud, and thus have probability values near 1. Pixels that are part of the land or water, but that have no cloud pixels near them, will have probability values near 0. Regions near the boundaries of clouds, or those that constitute heavy haze, will have probability values somewhere between 0 and 1. The scene registration process uses a previously stored model of the location of interest (port or airfield) in order to determine the precise location and orientation of the image. This facilitates the immediate labeling of a number of the areas within a scene. For example, knowing the precise location of the image makes possible the identification of fixed, invariant scene objects such as piers and runways. The use of this knowledge can facilitate more directed searches for objects and can provide supporting evidence of an object's presence or absence.

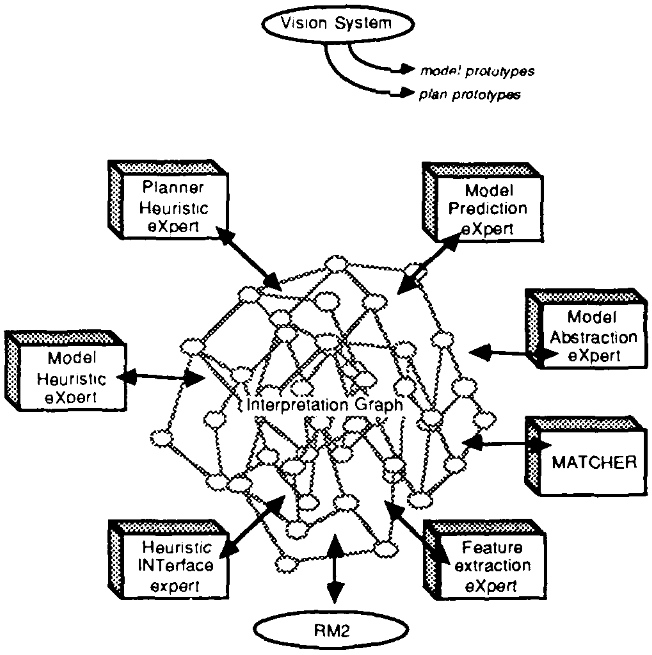


Figure 4 Symbolic Subsystem

The area level discrimination function combines the information produced by cloud detection and scene registration into a combined scene map. This map can then be queried by other discrimination levels to extract semantic information which can be used to guide processing. If the image is obscured and not enough of the image remains for subarea and region discrimination processing, the entire image will be discarded at this time. Otherwise, processing continues to the next level.

Subarea discrimination is the process by which portions of the image are identified to focus the processing of the vision system. These windows of interest are identified by either applying a pixel based operator to the image and generating blobs which have properties that have characteristics of man-made objects or by the use of contextual/historical information based on known scene registered locations.

Region level discrimination processing is the last step in the detection process to determine the presence of single or multiple man-made objects of interest and the rejection of clutter. Clutter regions can be generated by subarea processing. Such regions might represent ice floating in the water, smoke or small clouds that were not identified by cloud detection, or glint. During region level processing, they are rejected as clutter by performing a shape from shading analysis and by utilizing context in the reasoning process. In most cases, this leads to the elimination of such regions from further processing. Regions that do pass through this step, however, can still be identified as clutter and eliminated during the classification process based on more detailed shape matching and the search for object details.

Object level discrimination involves the hierarchical classification of objects as to class, subclass, and model. For example, an object in an airport which was passed by the region discrimination levels as being an airplane is analyzed to determine that the plane is a commercial aircraft (object class), due to, say, size, number of engines, and location of the aircraft on the runway. Information such as the above, in addition to knowledge of the wing sweep angle and other features, might allow further classification as a Boeing 747 (object subclass). Finally, fine detail might allow for identification of the aircraft as a 747-B (object model).

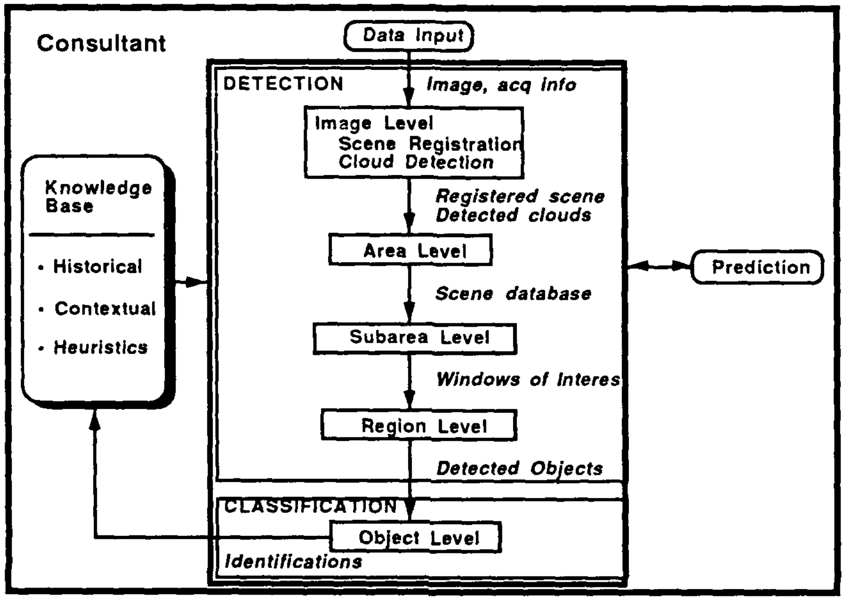


Figure 5 Overview of SCORPIUS discrimination levels

The discrimination levels are controlled by a meta-level component called the system consultant. The system consultant utilizes various knowledge bases and information provided by each of the discrimination levels to control and optimize system flow within and across discrimination levels. The other major system component which is accessed by the discrimination levels is the model prediction and mensuration system.

REGION AND OBJECT DISCRIMINATION LEVEL DETAILS

In this section further details on the specific approach to region and object level processing are presented.

The goal of region processing is to isolate single objects of interest while rejecting false alarms. The resulting orientation and position specifications are used by object discrimination to focus its search for predicted object parts. Another important design goal was to develop techniques that applied with a minimum of modification to two different scenarios, demonstrating that the SCORPIUS system could be extended to additional scenarios. A simple and powerful method has been developed for locating a class of generic cylinders [Binford, 1971] in images. Using constraint models and a flexible matching strategy, it has been applied successfully to both naval and airport facilities.

To a first approximation, the objects in which SCORPIUS is interested are cylinder shaped, or can be decomposed into cylinders. The spines of these cylinders are straight and the sweeping elements either maintain a relatively constant cross-section for most of the object (submarine and fuselage) or monotonically increase along its length (wings). Region processing accomplishes its task by finding cylinders of this restricted class in the image, then using contextual information to resolve the cylinders into submarines at a pier (seaport scenario) or into the wings and fuselage of an airplane (airport scenario). New scenarios will require the introduction of new generic models to guide processing. In the current system there are three algorithmic groups within region processing: fine registration, feature extraction and matching.

Fine registration which is invoked for the seaport scenario only, localizes the pier with the precision required to assign detected submarines to berthing zones. It uses the results of scene registration (image discrimination level) as an initial orientation and translation estimate. Using the Hough transform and projections, the required accuracy for both orientation and translation is achieved.

Rather than compute a complete shape from shading surface as in [Horn, 1970], the SCORPIUS system works with a sparse representation that facilitates the search for cylinders. Two methods of feature extraction are used to generate the sparse representation. Based upon the expected dimensions of the objects of interest, a set of spatial frequency bandpass images are created. Linear approximations of the ridges and valleys of the bandpass images form one set of features. The other set of features is derived from the topograghic primal sketch [Haralick, 1983]. Regions of pixels that are labeled convex and concave are skeletonized, and the linear approximations of these calculated. Figure 6 shows an idealized intensity profile and the relative locations of the skeletonized features along its length.

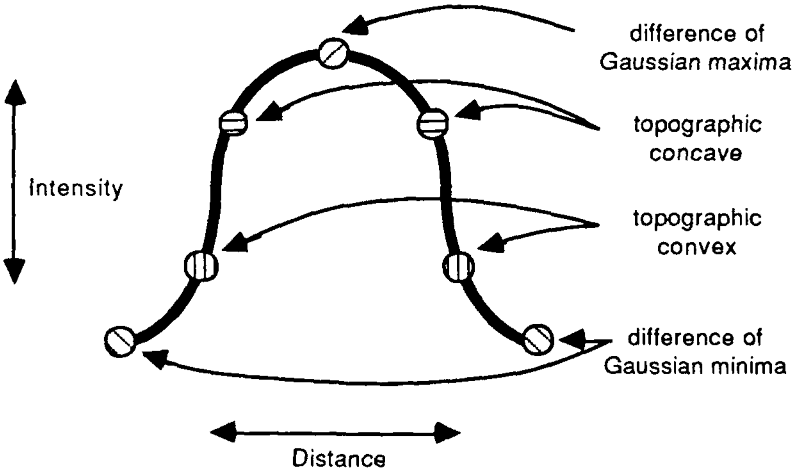


Figure 6. Arrangement of features across generic cylinder

A generic cylinder model specifies the order and perpendicular displacement among the feature lines that describe the cylinder. Since the perpendicular displacements among the lines vary with illumination and viewing angles, the generic model must also specify the acceptable range of values that can occur. In both scenarios, the cylinders are composed of 7 feature lines; however, the technique could easily be extended to accommodate other intensity profiles.

The matching process proceeds in two stages. First, it searches through the feature extraction results to find cylinders. Then, it combines the matched cylinders with any contextual information available for the pier or parking area to arrive at the proper configuration.

Cylinder matching begins by grouping the extracted feature lines into close, parallel bundles. Each bundle is bisected and the intersection point of the bisector with each feature line in the bundle is calculated. The intersection points provide the lateral displacement information used by the matcher. The matcher then applies the generic model to the intersection points of each bundle.

In the seaport scenario, several parallel cylinders can be significant, one for each submarine as well as one or two generated for the sides of the pier. In the airport scenario, close, parallel cylinders are not expected. For this reason, the matcher used by the seaport scenario must return as many non-overlapping cylinders as it finds, while the airport scenario matcher can simply return the one best match.

The final step in the process is to generate a configuration from the cylinders. Employing knowledge of the pier location and width, the seaport scenario configuration matcher removes the cylinders that are artifacts of the pier, then assigns berthing zones to the remaining cylinders. In the airport scenario, a simple set of length and angular constraints is used to form an airplane. Should one of the components of the airplane be missed by the cylinder formation process, the configuration matcher can hypothesize the missing component. When the fuselage is the missing component, the angular relationship between the wings provides unambiguous evidence to complete the configuration. If a wing is missing, then evidence of the wing inserting into the fuselage and/or contextual information provided by registered scene model is used to complete the configuration.

Object processing is a model driven fitting of stored and calculated expectations to evidence extracted from the image. Using available historical and contextual data for the parking location under consideration, a set of possible identifications is generated. 2-D projections of stored wire-frame models are generated for each candidate identification at the orientation specified by region processing. Specific measurable appearance data, by which object parts can be located and relationships among the parts compared, is derived from the 2-D projections. Possible part locations are identified using the appearance data for each part. The matching step generates a best fit between the possible part locations and the expected spatial relationships among the parts, yielding a confidence value for each candidate identification.

Image morphology forms the basis of the part location capability for the object discrimination level. Using a combination of grey-scale and binary structuring elements, object parts can be emphasized and located with a high degree of specificity. The grey-scale structuring elements are applied as a preprocessing step and are parameterized on the basis of image statistics, resolution and object size. The binary structuring elements are dynamically created specifically for each of the object parts and are applied to the preprocessed image to generate possible part location.

Objects are represented within the system in several forms. Aside from a 3-D wireframe model, objects are also represented as a set of part frames which contain image-invariant information. In order to custom generate structuring elements for the object parts, the 3-D wireframe models of the candidate set are predicted at the orientation specified by region discrimination processing. The prediction is warped to match the taking conditions of the image, therefore, statistics calculated for modeled parts reflect those of the imaged parts. By combining the image-invariant information from the part frames with the prediction derived statistics, a set of image specific structuring elements is created for each part. Another set of frames associated with an object are called discriminants. These are used to record the distance/orientation relations among all the object's parts for later use by the matcher.

The most specific structuring element for a part is a binary silhouette of the part. This 'literal' structuring element is effective only for parts whose intensity signatures are relatively constant across their extent. For many cylinder-like parts, a 'linear' structuring element is effective. While this class of part is usually wider than a pixel, its intensity signature forms a ridge that is most amenable to enhancement by a structuring element with a single pixel width. Finally, for parts whose intensity signature is dominated by a small patch of specular reflectance, or whose structure is below the resolution of the image, no structuring element is generated and the parts are matched against bright spots extracted from the image.

Once the prediction, structuring element generation and preprocessing have completed, the part extraction process begins. A backward chaining ruleset attempts to find object parts in the image, using progressively less specific structuring elements. A part is considered to have been found when one or more hypothesized locations passes a test that is both part and scenario specific. The shape, size and orientation of the hypothesized part location must be within acceptable limits, and using the results of region discrimination, the part must be located in a particular relation to the configuration. For example, to be a left wing engine, the hypothesized part must be parallel to the fuselage, within acceptable length limits and proximal to where the left wing has been localized. All part hypotheses that pass their test are passed on to the matcher.

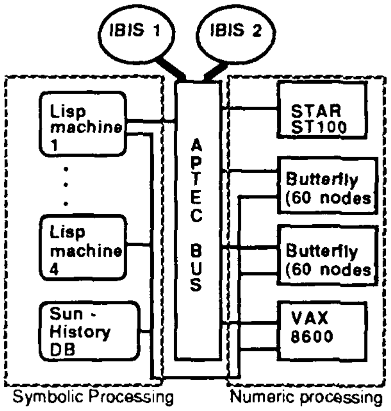
Input to the matcher are the possible part locations extracted from the image and the pairwise distance/orientation measurements taken from the 2-D projection of a single object. It performs a depth first search of the possible combinations of hypothesized parts to determine the best fit. A numerical confidence value is assigned to the degree of fit between the hypothesized parts and the model. The match step is repeated for each candidate in the set.

EXPERIMENTAL PROTOTYPE SYSTEM

The experimental prototype system (EPS) is the heterogeneous parallel processing testbed used to implement the SCORPIUS vision system. The testbed has evolved from a single user system to a multi-user system in order to better support the development and testing activities of the program. The current architecture of the EPS is shown in Figure 7. Some of the key differences from the original architecture include splitting the 120 node Butterfly machine into two 60 node machines with each connected via a high speed interface to the Aptec bus. This change provides system redundancy and physical hardware separation between users since the Butterfly Chrysalis operating system has limited memory protection. A sophisticated RAMfile system for the Butterfly machine was developed to suppo' 'he handling of large images at high data transfer rates across the Aptec bus into and out of the Butterfly machine transparent to the application programmer. A second parallel transfer IBIS disk subsystem is being added for system redundancy and for additional disk space. A Vax 8600 is being used instead of the Vax 750 for increased computational power. The single lisp machine with a Butterfly machine acting as a backend processor has been replaced by multiple lisp machines each which can individually access the testbed. The backend 16 processor Butterfly machine based on the BF1 model was eliminated due to address space limitations and the poor performance of a beta version of parallel lisp based on Scheme. At the current time, no shared memory parallel lisp implementation alternative is commercially available. A Sun workstation based relational database to support history and collateral data has been added and will serve as a source of validated information that can be used to support the operation of the vision system and to save processing results. The Warp machine which was included in the original architecture is not being integrated into the testbed due to programmatic constraints. The STAR ST-100 array processor which has similar capabilities to the Warp has been integrated. A number of major enhancements to the Warp hardware and software environment were worked jointly with Carnegie Mellon University and General Electric. The DARPA Intacts program which has adopted a similar architecture as SCORPIUS will be completing the integration of the Warp into their system.

DEVELOPMENT STATUS & PLANS

Table 1 summaries the various major prototypes and milestones of the SCORPIUS program. Of the prototypes listed three major milestones remain to be completed and include P4 vision system demonstration (August 1989), Experimental Prototype System acceptance test (November 1989) and EPS test and evaluation (March 1990). The enhanced P2 vision system is currently undergoing test and evaluation on 16 images from two different scenarios and includes four target locations. In parallel, additional capabilities are being developed for integration into the next P4 vision system prototype demonstration. The P4 demonstration will include the first fully integrated used of the parallel processing equipment. After the P4 demonstration, additional algorithms are planned to be moved to the parallel processing hardware and the system will undergo further integration and test resulting in an acceptance test. Once accepted, the system will be tested and evaluated using a database of 100 images of varying complexities and will be documented in an extensive evaluation report.



Features:

- Development mode on target machines
- Integrated heterogenous processing environment
- 2.8 Gigabytes of disk storage (access at rates up to 9 Mbytes per sec)
- 3.6 Gigabytes of VAX disk storage
- Support for up to 4 users doing integrated system testing

Figure 7. SCORPIUS Multi-user Parallel Processing Testbed

The prototyping approach used on the SCORPIUS program has proved to be an extremely useful approach in structuring the research and development activities of the program. Utilizing this approach, the SCORPIUS program has evolved in a flexible manner, has resulted in a better problem understanding and has provided the customer with valuable information related to long lead time research and development activities needing further effort.

ACKNOWLEDGEMENTS

The work described in this paper is based on the efforts of many individuals. The key contributors include R. Cody, W. Diamond, J. Edwards, D. Finn, S. Gee, H. Iwata, G. Maruyama, A. Newman, R. Onishi, W. Osbon, A. Othon, P. Poduska, B. Schara, G. Tucker, R. Vonesh, R. Wheeler, C. Wilson, B. Moxon (BBN) and J. Web (CMU).

REFERENCES

1. "SCORPIUS Program Plan," Version 2.0, Hughes Aircraft Company, September 1987

2. J. F. Bogdanowicz, "Image Understanding utilizing Strategic Computing Initiative Architectures," SPIE Vol 758, Image Understanding and The Man-Machine Interface, 1987, pp.60-68

3. B. W. Boehm, " A Spiral Mode! of Software development and Enhancement," Computer, May 1988, pp 61-78

4. M. Berlin, J. F. Bogdanowicz and W. Diamond, "Planning and Control Aspects of the SCORPIUS Vision System," DARPA Knowledge-Based Planning Workshop, December 1987, Austin, Texas

5. T. O. Binford, "Visual Perception by Computer," presented to the IEEE Conference on Systems and Control, December 1971.

6. B. K. P. Horn, "Shape from shading: a method for obtaining the shape of a smooth opaque object from one view," Technical Report MAC-TR-79, Project MAC, Massachusetts Institute of Technology, 1970.

7. R. M Haralick, "The Topographic Primal Sketch," The International Journal of Robotics Research, Vol. 2, No. 1, Spring 1983, pp 50-72.

Table 1. Summary of SCORPIUS Prototypes and Key Milestones

Prototype/Milestone	Purpose	Date
P0	Process new test imagery, current algos	July 1986
P1	Enhanced algos, new vision system architecture	April 1987
Enhanced P1	Algos integrated into enhanced vision architecture	Nov 1987
P2	Major restructuring of vision system, enhanced vision architecture, new algo approaches, 2 scenarios, benchmarks on Butterfly, Warp, & STAR	April 1988
Enhanced P2	Further enhancements for speed & robustness (process 4 integration & test images)	Dec 1988
P2 Test & Evaluation	Process 18 images: 4 different sites, 2 scenarios	April 1989
P3	Demo of system interfaces	Sept 1987
P4	Further image complexities & improved speed using parallel processing testbed	Aug 1989
P5	Integrated demo of interfaces & distributed exec	Jan 1988
EPS Acceptance Test	Final test of EPS prior to test & evaluation	Nov 1989
EPS Test & Eval	Process up to 100 images	March 1990

Note: Underlined dates indicate milestones yet to be completed.

Scene Registration in Aerial Image Analysis

Frederic P. Perlant and David M. McKeown
Digital Mapping Laboratory
School of Computer Science
Carnegie Mellon University
Pittsburgh, PA 15213

Abstract

In this paper we discuss the importance of scene registration for several tasks in the automated interpretation of aerial imagery. These tasks are structure matching, stereo matching, and stereo visualization. While the processes of registration and matching have traditionally been treated as separate problems, particularly in the case of stereo matching, we describe techniques that may unify these processes. We also demonstrate the automatic generation and matching of control points in complex aerial imagery and show that the resulting registration is comparable to that achieved using manual control point selection. Finally, methods for the generation and visualization of stereo disparity images and stereo ground truth scene segmentations are described.[1]

1. Introduction

Scene registration is a fundamental requirement for a number of image analysis tasks such as stereo matching, multi-image matching for temporal changes, and image sequence or motion analysis. As a result there exists a rich variety of techniques to perform scene registration. For example scene registration can be accomplished by identification of image points to a common frame-of-reference via control points whose three-dimensional location is accurately known. Registration can also be accomplished in a relative manner by identifying corresponding points between one or more images, i.e., the establishment of image-to-image control points. The position of these points need not to be known in the three-dimensional world. For some applications registration can be accomplished with respect to a cartographic map, a photomosaic, or an orthophoto that has been warped in order to remove position distortions due to terrain relief.

Depending on the type of registration there arise many issues in accuracy. Accuracy is generally evaluated within the context of a particular task requirement. Often techniques with some inherent inaccuracy can suffice in many task situations. If we look at traditional photogrammetric techniques for recovering the position of an image we require detailed information of the camera, its focal length and lens characteristics, the platform coordinates in terms of height above the ground and its three-dimensional position (pitch and yaw). Inaccuracies arise when we are unable to know any of these parameters precisely. In the case of digital imagery, the image formation process can introduce additional errors. For example, the sampling process, in terms of geometric precision and radiometric accuracy, is rarely modeled for digital imagery generated by digitizing photographic film. There are a variety of digitization apparatus including rotating drum scanners, flatbed optical scanners with a single optical element, scanners composed of linear arrays of elements or a full three-dimensional element array. Each has its own inherent inaccuracy. The kinds of errors introduced include those due to sampling a continuous tone photograph into a discrete intensity range over an arbitrary sampling window and limitations in the sensors' dynamic response. The geometric accuracy with which the scanner is positioned and moved over the film varies greatly between scanning methods. Even once we have accurate image, sensor, and platform information our ability to locate ground control points accurately in digital imagery is independent of the inherent accuracy of those control points.

This paper raises issues in how scene registration can be achieved in digital imagery and illustrates the importance of accurate registration for three analysis tasks. In Section 2 we discuss some general issues in registration in computational vision. These issues include use of a spatial database to provide coarse ground control information, the selection of manual control points, and the automatic determination of control points. In Section 3 we discuss the importance of registration to three particular tasks in the interpretation of aerial imagery.

[1] This research was primarily sponsored by the U.S. Army Engineering Topographic Laboratories under Contract DACA72-87-C-0001 and partially supported by the Defense Advanced Research Projects Agency, DoD, through DARPA order 4976, and monitored by the Air Force Avionics Laboratory Under Contract F33615-87-C-1499. The views and conclusions contained in this document are those of the authors and should not be interpreted as representing the official policies, either expressed or implied, of the U.S. Army Engineering Topographic Laboratories, or the Defense Advanced Research Projects Agency, or of the United States Government.

These tasks are the correlation (or fusion) of monocular analysis from partially overlapping views, computational stereo matching techniques, and the visualization of stereo matching results. We also discuss the interrelationship between registration and matching. In Section 4 we describe the results of a fully automatic scene registration from initial coarsely registered stereo pair to a final three-dimensional interpretation. Finally, in Section 5, we present conclusions and discuss future directions for our research.

2. Scene Registration

In this section we describe some basic principles of stereo photogrammetry, a part of computational photogrammetry [25]. The primary goal of stereo photogrammetry is to determine the three-dimensional position of any object point that is located in the overlap area of two images taken from two different camera positions. The determination of the orientation of each camera at the moment of exposure and the relationship between the cameras is a necessary step in the photogrammetric process. In Section 2.1 we discuss the problem of camera orientation that determines the relationship between the image points and ground points in the scene. In Section 2.2 we describe the classical epipolar geometry for stereo imagery. When two images are registered in the epipolar geometry the spatial relationship between corresponding points in the left and right images is greatly simplified.

2.1. Camera orientation problem

The solution to the general camera orientation problem has four components: the interior orientation, the exterior orientation, the relative orientation, and the absolute orientation.

The *interior orientation* refers to the perspective geometry of the camera. The parameters of the camera are generally known *a priori* and can be determined by precise calibration. This includes the focal length, the position of the principal point in the image plane of the camera and the geometric distortion characteristics of the lens system. These parameters are intrinsic to the camera and are generally detailed on a standard camera certificate.

The *exterior orientation* characterizes the orientation of the camera during the image event. It is defined by the geographic position of the optical center in a three-dimensional rectangular coordinate system and the direction of the optical axis. Therefore, the exterior orientation determines the projective relationship that exists between the image coordinates of the image points and the ground coordinates of the corresponding object points in the scene. In the context of stereo photogrammetry, the exterior orientation can be decomposed into the relative orientation and the absolute orientation.

The *relative orientation* determines the relative three-dimensional position of the two images in the stereo pair with respect to each other. As shown in Figure 2-1 three lines characterize the exterior orientation of the two cameras. Two lines are the rays emanating from an object point P and passing through each of the optical centers, C_L and C_R, of the two cameras. These lines are P to P_L (passing through C_L) and P to P_R (passing through C_R). The third line is C_L to C_R passing through each of the optical centers. This is called the baseline of the stereo model. These three lines are represented in a three-dimensional rectangular coordinate system and must be coplanar. This coplanarity relationship gives an equation with twelve parameters that defines the exterior orientation of the two cameras. Of the twelve parameters, just five are necessary to define the relative orientation of the two cameras. In order to determine the five parameters, we need five pairs of corresponding points (P_L,P_R) in the left and right image. Each left/right pair defines one coplanarity equation.

After the relative orientation is accomplished, the stereo model must be scaled, translated, and leveled with respect to a ground reference system. The process of orienting a stereo model into an absolute reference system is called the *absolute orientation*. It relates the absolute coordinates of an object point in the ground reference system to its coordinates in the model coordinate system of the camera. Each control point, for which the ground coordinates are known, gives rise to three projective transformation equations when its model coordinates are measured. Three control points are necessary to define all the parameters of the absolute orientation.

In our work we consider that the interior orientation or calibration has been already performed since we assume an ideal pinhole camera. Therefore, the calculation of the relative orientation is the registration problem [12]. Knowing the relative position and attitude of the two images in the stereoscopic pair with respect to each other defines the relationship between the two images of the scene. All of the results presented in this paper will be relative measurements. However, as we have discussed, these relative measurements could be used to calculate absolute metrics, such as height, length, and area by using three-dimensional ground control points to establish the absolute orientation.

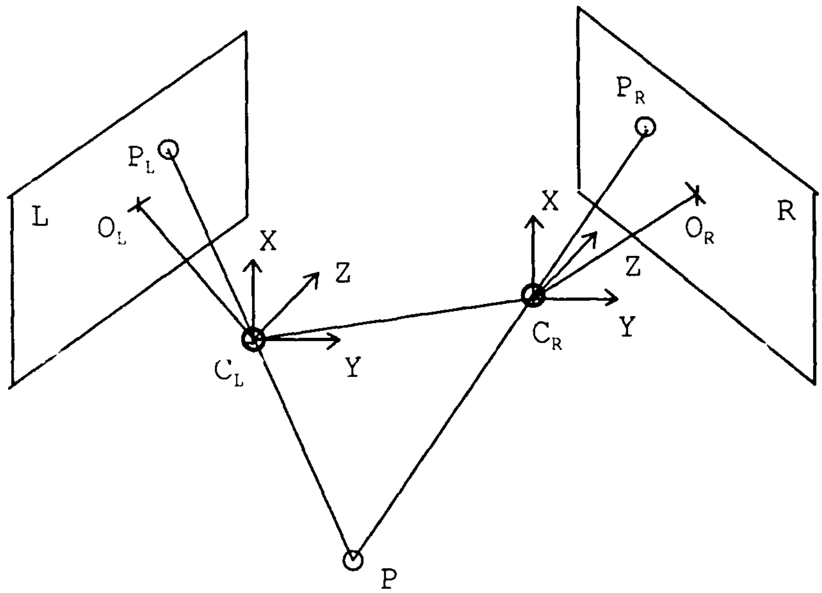


Figure 2-1: Relative orientation of stereo imagery

2.2. Epipolar geometry

After we have established the relative orientation of the two images, it is possible to reformulate this relationship into the epipolar geometry. The *epipolar geometry* defines a constraint based on the geometric relationship that the plane containing the two optical centers of the cameras, C_L and C_R, and the ground point P, intersect the two image planes on two lines. These lines are called the conjugated epipolar lines. As shown in Figure 2-2 these lines, E_L to P_L and E_R to P_R, contain the two image points P_L and P_R. These lines emanate from a common epipolar center, E_L and E_R. This epipolar center corresponds to the intersection of the image plane and the stereo baseline C_L to C_R. Thus, the points in the left and right image (P_L and P_R) correspond to a single three-dimensional scene point (P) and are on the same conjugate epipolar lines E_L to P_L and E_R to P_R. After this relationship has been established, it is common to register the two images so that the conjugated epipolar lines become corresponding scanlines in the left and right image. Therefore the corresponding points are on the same scanline in each image and the displacement between the points, or disparity, corresponds to the relative height of the three-dimensional scene point.

This epipolar geometry is a common framework for most stereo matching algorithms: [1], [5], [6], [22], [24]. These stereo matching techniques assume that the registration is ideal and that the epipolar constraint is completely satisfied. Some researchers have attempted to explicitly account for the inaccuracy of the image registration and have attempted to improve it by preprocessing the imagery before beginning the matching process [8], [9], [11], [29]. Modeling inaccuracy in image registration has most often been studied within the context of the robotic applications where it is common to have a good deal of control over the cameras [10] and where a detailed preregistration and calibration step is possible. However, for many applications in aerial image analysis one is often simply given overlapping images or partial image areas where the epipolar geometry must be derived.

In the following section we present two methods for scene registration given overlapping stereo imagery. The first method performs a coarse registration using landmarks from a spatial database. The second method uses pairs of corresponding points in the two images to perform a relative orientation. As we will see, many of the techniques used in computer vision to establish scene registration are approximations to the photogrammetric ideal. These approximations cause the scene registration to be inaccurate. The effect and implications of these inaccuracies will be explored within the context of two matching tasks.

2.3. Coarse registration using a spatial database

The most common method to establish the relative orientation between two images is to select pairs of corresponding points in the two images. One alternative method is to independently tie each image to a common frame of reference. A cartographic coordinate system such as <latitude,longitude,elevation> is one possible frame of reference. Thus, the two images are related to a ground coordinate system, or map. The use of

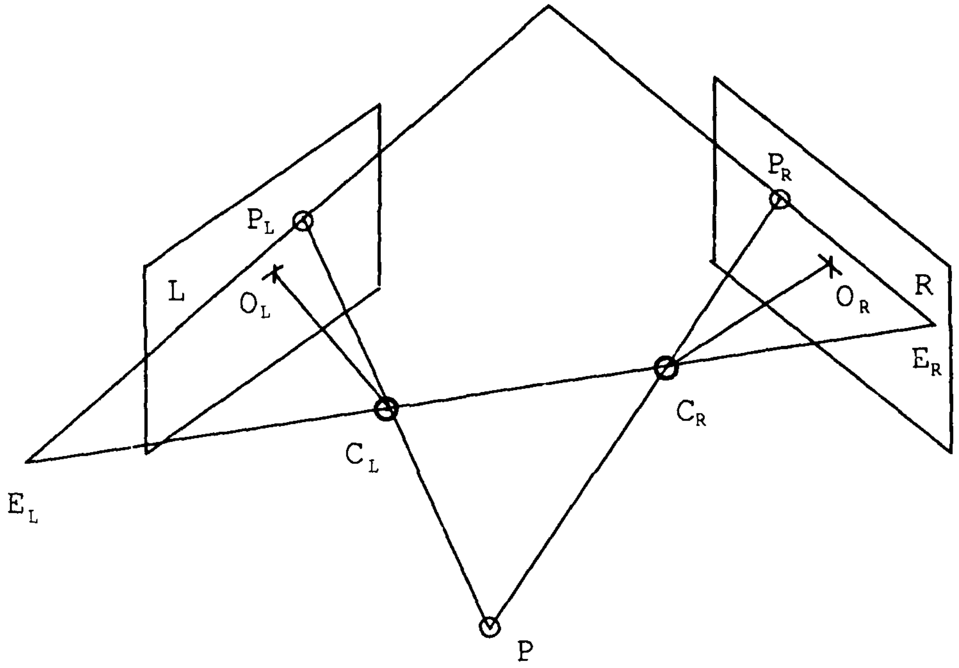


Figure 2-2: The epipolar geometry

landmarks with known <latitude,longitude,elevation> is a common method to orient each image. The overall accuracy of the registration is dependent on the accuracy of the three-dimensional position of the landmark and the accuracy with which we can recover the image position of the landmark. We use the landmark database component of CONCEPTMAP, a spatial database system that integrates imagery, terrain, and map data to provide landmark descriptions [17, 18]. Each landmark description in the database has a reference image fragment, and a ground position definition which contains the <latitude,longitude,elevation> information, its position in the reference image fragment, and a brief textual description of the landmark for the user. Each image in the CONCEPTMAP database is put into correspondence using manual selection of landmarks.

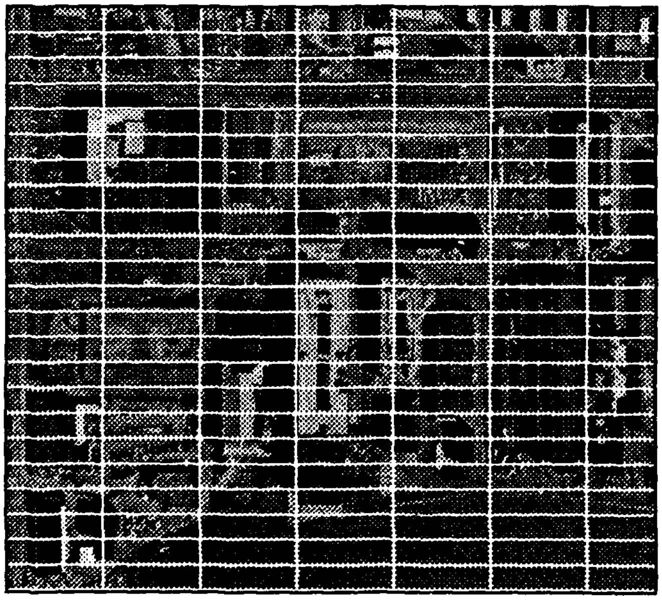


Figure 2-3: Left image DC38008 with CONCEPTMAP database registration

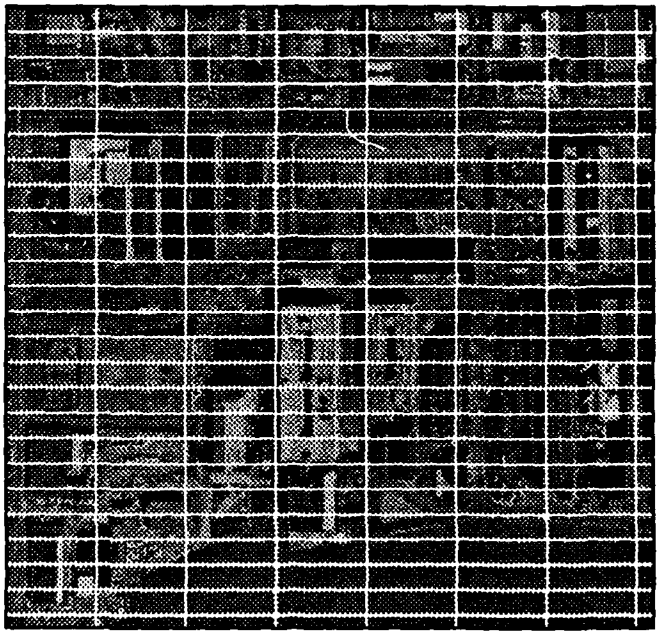


Figure 2-4: Right image DC38007 with CONCEPTMAP database registration

Figure 2-3 and 2-4 show a stereo image pair of an industrial area taken from the CONCEPTMAP database. These images were digitized from standard nine inch format mapping photography taken at the altitude of 2000 meters by a 153 millimeters camera. One pixel corresponds to 1.3 meters on the ground. The left image is a 512 x 512 sub-area selected from 2300 x 2300 image. The right image sub-area was generated by calculating the <latitude,longitude> for the corner points of the left image and projecting those points onto the complete right image. This projection is then used to extract the image sub-area from the complete right image. We have superimposed a set of gridlines on both images in order to make it easier to see the actual misregistration. Typically CONCEPTMAP provides a registration accuracy of between ten to thirty meters for imagery digitized to a 1.3 meter ground sample distance.

2.4. Fine registration using image control points

As we have seen, the computation of the relative orientation can be accomplished by selecting pairs of corresponding points in the two images. After the relative orientation is calculated, two images can be transformed so that they satisfy the epipolar constraint. We begin the fine registration with the coarse registration described in the previous section.

We make several assumptions that simplify the relative orientation model. We assume that the cameras are metric and have the same interior orientation. We also assume that their optical axes are parallel and are, in fact, vertical. Because we are using aerial imagery taken by the same camera along the same flightline, these assumptions are not unreasonable. The largest source of error is whether the camera platforms were at precisely the same altitude and orientation at each imaging event. Given these assumptions, the transformation between the left and right image is only a translation and a rotation, because the image planes are the same. Therefore, of the five parameters describing the general relative orientation, only two are remain to be solved. The absolute distances in the two images are preserved and the epipolar lines are already parallel. After the transformation the epipolar lines correspond to the scanlines.

Problems with the accuracy of point selection lead us to use more points to determine the transformation between the two images. On some images this model was not flexible enough to account for the variation in the ground elevation and the variations to our ideal sensor model. As a result we developed a polynomial transformation adjusted by least squares to fit the selected corresponding points. In the following sections we discuss the manual and automatic selection of image control points.

2.4.1. Manual selection of common points

The classical method to select corresponding points in order to perform interior orientation is by the manual identification of landmark points in stereo imagery. Typically, man-made features such as road intersections, boundary corners of fields or parking lots, or markings such as road centerlines are used because of the ease with which they can be found in the imagery. We chose to manually select shadow corners since these points were the focus of our experimentation in automatic landmark detection. Given that we are working in an urban environment, shadow corners have the advantage that they are generally on the ground and therefore in the same plane, assuming only small changes in terrain elevation. Although the shadow position changes as the sun moves, if we have imagery taken at nearly the same time, as is common in aerial mapping photography, the shadow corners will fall on the same point in the three-dimensional scene. Such corners also tend to be uniformly distributed in scenes containing large numbers of buildings. The manually selected shadow corners give us a baseline against which we could measure the accuracy of the automatic landmark selection process. Figures 2-5 and 2-6 show the manually selected shadow corners in the left and right images respectively.

2.4.2. Automatic selection using shadow corners

Clearly, one requirement for automated registration is the automatic selection of corresponding points in the stereo pair images. There are actually two problems that must be solved. First we must automatically detect potential landmarks in each image, and then we must determine those landmarks that have been found in both images. General landmark matching is an unsolved problem and most automatic registration techniques relie on the matching of characteristic points [21] that often have no physical significance or reference with respect to landmarks.

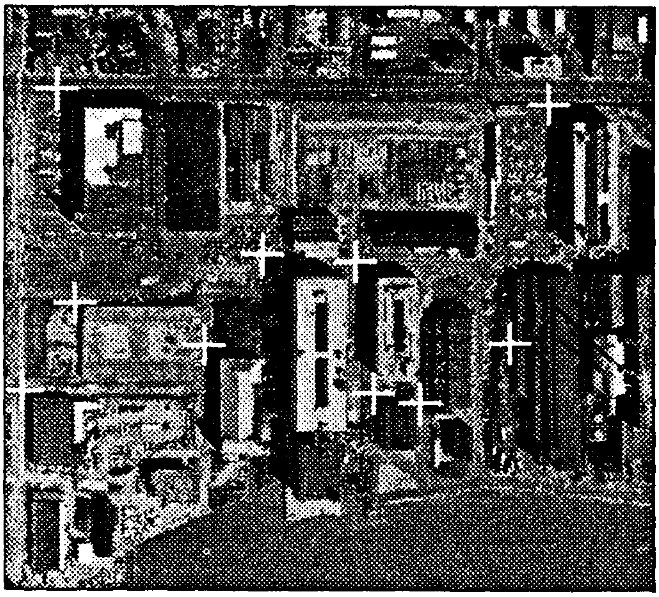


Figure 2-5: Manual selection of points [left image] with coarse registration

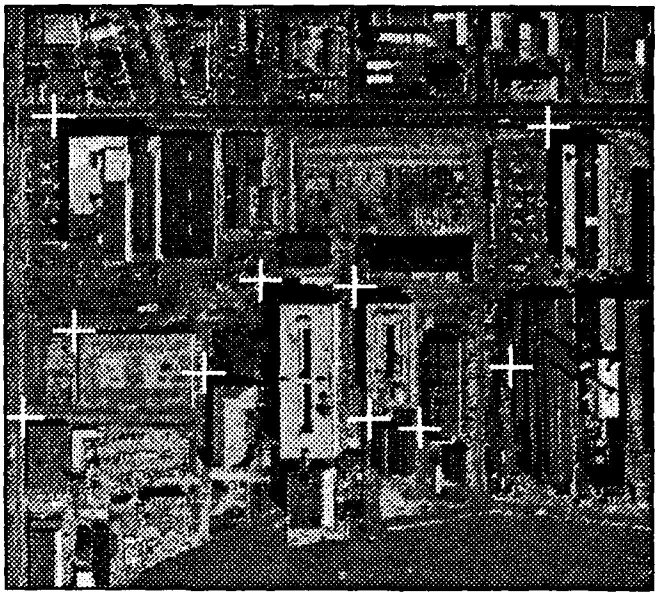


Figure 2-6: Manual selection of points [right image] with coarse registration

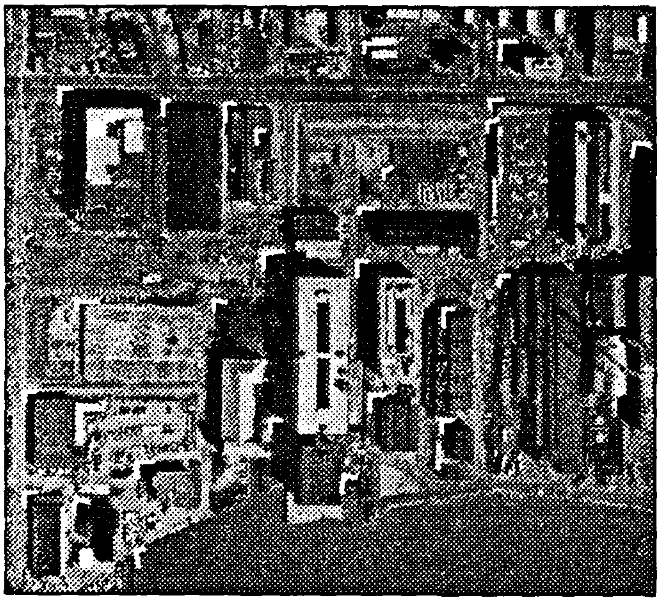


Figure 2-7: Automatic selection of points [left image] with coarse registration

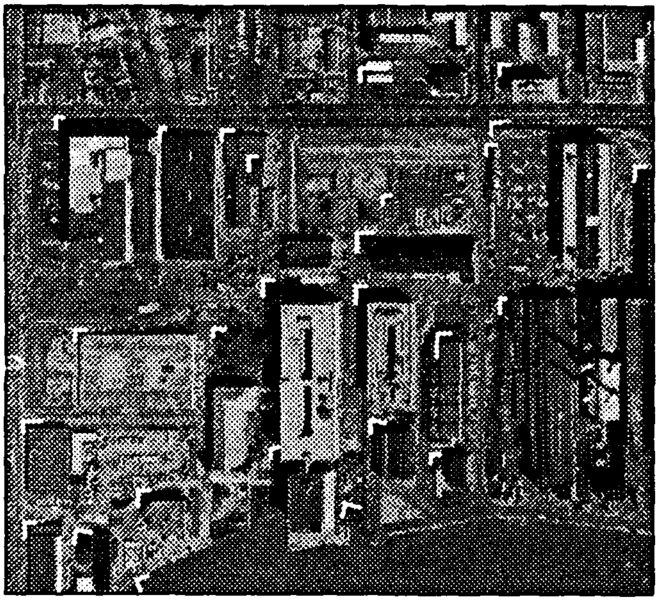


Figure 2-8: Automatic selection of points [right image] with coarse registration

For this experiment, we assume that a coarse registration of the two images, such as described in Section 2.3 has already been performed. Using this coarse correspondence, we are able to limit the search to find corresponding features in the images. Most of the remaining error is translational rather than rotational which simplifies the determination of corresponding points.

As described in the previous section, shadow corners are good candidates for automatic detection and correspondence as well as for manual selection. We use a monocular detection of shadow regions and determine the boundary line between the shadow and the building [14]. This boundary is used to determine the position of the shadow corner in the left and the right images [2]. After removing corners that were inconsistent with shape and orientation constraints imposed by the sun angle, we selected sets of shadow corners that were detected in both images. Figures 2-7 and 2-8 show these corresponding shadow corners on the two images. Note that the corners selected differ from those selected manually.

Figures 2-9 and 2-10 show the results of the fine registration using shadow points selected automatically. This registration is obviously better than the coarse registration using the CONCEPTMAP database shown previously in Figures 2-3 and 2-4. In the following section we attempt to quantify the registration quality.

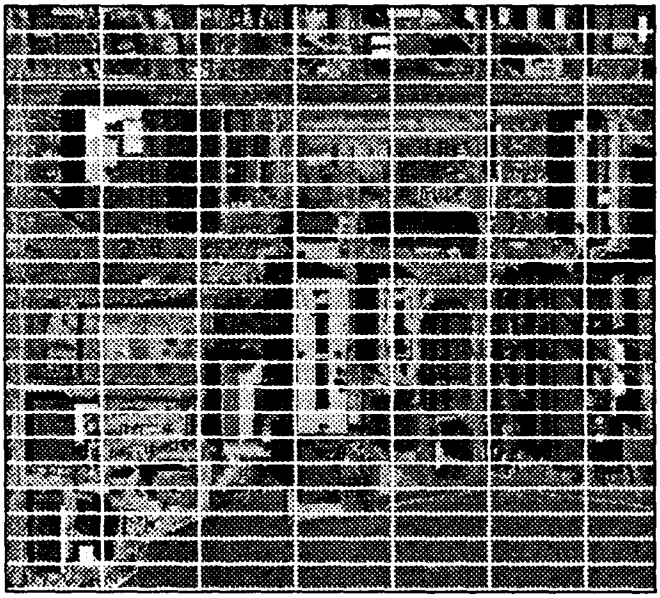


Figure 2-9: Left image of the fine registration

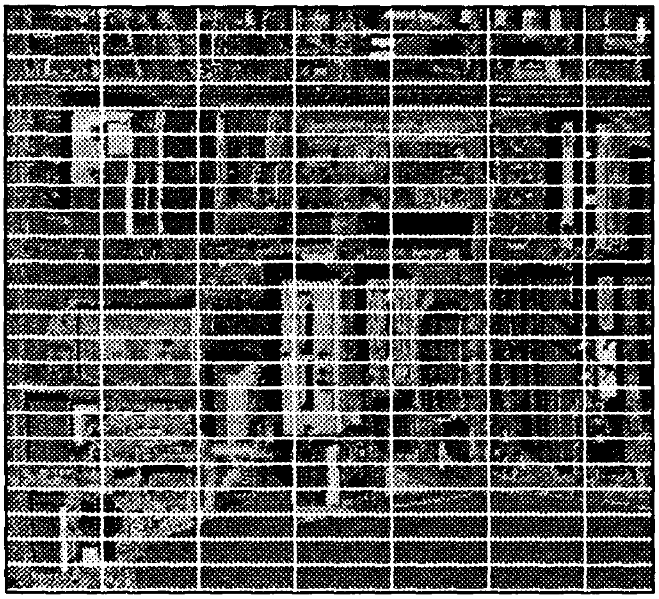


Figure 2-10: Right image of the fine registration

2.4.3. Quality of registration

Tables 2-1 and 2-2 show the local accuracy of the different scene registrations performed on DC38008 and LAX stereo image pairs.[2] The first three rows of each table characterize the quality of the CONCEPTMAP registration using three set of control points: the points selected manually, the points generated by automatic detection of shadow corners, and the points derived from structure matching. In the case of DC38008 11 conjugate point pairs were manually selected, 26 shadow corner pairs were automatically extracted, and 16 point pairs were found using structure matching. In the case of the LAX stereo pairs, these numbers are 14, 13, and 16, respectively. Because the CONCEPTMAP coarse registration is derived by a polynomial fit for the entire scene (2300x2300 for DC38008 and 2000x4000 for LAX), it is interesting to evaluate the quality of the local fit for the 512x512 image sub-areas using each set of independently derived control points. The CONCEPTMAP registration produced a translation to within approximately 12 pixels (16 meters) of the the true registration for both images. This registration was quite consistent across all three sets of test points.

For the other two types of scene registration, isometrical and polynomial, we evaluated the quality of registration with respect to the manually derived control points. That is, the solutions for manual, corner, and structure matching were validated using the manual points. In all cases both registrations achieved significantly better results than the CONCEPTMAP coarse registration. In several cases the registrations achieved by matching shadow corners and structures is quite comparable to the manual registration. However, the manual registration is in all cases as good as any of the automatic control point experiments. In all cases the manual selection of corresponding points produced a registration of less than one meter, or subpixel accuracy. In some cases similar subpixel results were achieved using the automatic point selection. Finally, the polynomial approach led to better results although the simpler isometrical model gave comparable results.

One additional issue is how well our local solution performs as a global registration in other areas of the complete stereo pair. In Table 2-3 we show the results of using our local fine registration for both the isometrical and polynomial methods in four quadrants of the complete stereo pair. In each of the four quadrants we manually selected 12 control points and used the manual solutions for DC38008 to calculate residual errors. Because of the large variation in the row and column offsets, it is clear that the local model can not be treated as a global model even though the row residuals stay within reasonable bounds. However, it is the case that the fine solution should be a better global solution than the CONCEPTMAP coarse registration.

[2] *We will introduce some matching results for the LAX airport scene in Section 3.1.2.*

Statistics on the quality of the different registrations for DC38008						
Type of registration	Number of points	Avg. row offset	Std. row offset	Min/Max row off.	Avg. col offset	Std. col offset
Coarse manual	11	-12.4	1.6	-15/-8	905.5	1.2
Coarse corner	26	-13.2	1.6	-18/-10	905.7	1.7
Coarse structure	16	-12.1	1.6	-15/-8	909.3	3.4
ISO manual	11	0.1	0.7	-1/2	1.3	1.3
ISO corner	11	1.7	1.7	-1/7	5.1	1.2
ISO structure	11	0.5	0.6	0/2	-3.4	1.4
POLY manual	11	0.1	0.3	-1/1	0.1	0.5
POLY corner	11	-0.2	1.8	-3/4	0.0	1.6
POLY structure	11	0.1	0.5	-1/1	-3.3	1.5

Table 2-1: Statistics for different registrations on DC38008 stereo pair

Statistics on the quality of the different registrations for LAX						
Type of registration	Number of Points	Avg. row offset	Std. row offset	Min/Max row off.	Avg. col offset	Std. col offset
Coarse manual	14	10.6	0.9	9/13	1866.6	0.7
Coarse corner	13	10.9	0.7	9/12	1866.8	1.4
Coarse structure	16	10.9	0.4	10/12	1869.3	1.7
ISO manual	14	-0.4	0.9	-2/2	0.6	0.7
ISO corner	14	-0.4	0.9	-2/2	1.6	0.7
ISO structure	14	-0.4	0.9	-2/2	-2.4	0.7
POLY manual	14	0.0	0.1	-1/1	0.1	0.7
POLY corner	14	1.3	1.0	-1/3	1.5	0.9
POLY structure	14	-0.3	0.7	-1/2	-2.9	0.9

Table 2-2: Statistics for different registrations on LAX stereo pair

Quality of the registrations for the complete image DC38008						
Type of registration	Region in image	Avg. row offset	Std. row offset	Min/Max row off.	Avg. col offset	Std. col offset
ISO manual	North	1.7	1.5	0/4	4.5	0.8
	West	-1.3	0.4	-2/-1	1.4	1.2
	East	1.3	0.6	0/2	-2.4	1.5
	South	-2.5	0.8	-5/-1	3.1	1.1
POLY manual	North	-1.6	1.2	-4/0	-70.5	16.6
	West	-1.3	0.7	-2/0	-2.4	3.0
	East	-0.1	0.3	-1/1	2.7	3.7
	South	0.8	0.7	-2/2	-51.5	14.3

Table 2-3: Statistics for different registrations on DC38008 stereo pair

Although traditional error analysis can give us an idea of relative accuracy for each of these approaches, this does not necessarily translate into the effectiveness of the registration. That is, for many tasks in scene analysis a coarse-grain registration to within 10 to 30 meters is quite adequate, especially considering that the imagery covers several square kilometers. For instance, tasks that require assembling a collection of image subareas taken over time for change detection and analysis can be supported using this level of accuracy. However, for other tasks, such as matching and stereo analysis, the effect of mis-registration may become more critical. In Section 3 we will see how such tasks are affected by coarse and fine registration.

3. Tasks Requiring Accurate Scene Registration

In this Section we describe three scene analysis tasks that require or support scene registration. These tasks are matching structures derived by monocular analysis of overlapping imagery, traditional stereo matching using area-based and feature-based matching techniques, and the construction of a three-dimensional image to present matching results to a human using a stereo display monitor.

In the case of matching of monocular structures, we can acquire additional information about the actual structure of the objects, including their height, as a result of the matching process, and also generate new automatic control points to refine the registration. The goal is to match high-level structures in two overlapping images, where the actual detection and delineation of the structures is likely to contain significant errors, and matching is complicated due to a large numbers of false alarms produced by the structure generation process.

In stereo analysis the goal is to automatically match points in the left and right images of the stereopair in order to establish a *disparity* between these points. This disparity can be used, along with the camera model, to calculate the actual height of the matched point in the three-dimensional scene.

Finally, it is becoming increasingly important for researchers to be able to visualize the three-dimensional models that their analysis programs are generating. Such a visualization tool allows us to directly compare these results to three-dimensional ground truth models for performance evaluation.

3.1. Correlation of monocular analysis

There are many situations where overlapping coverage imagery is available but may not be suitable for stereo matching due to sensor acquisition parameters, temporal or seasonal changes, or image scale. The issue then becomes one of how to relate the results of independent monocular analysis. One of the first examples in the literature was symbolic change detection [26, 27] and the matching of coarse regions such as lakes, fields, and forests based upon relationships that were largely invariant over small rotations in the image plane (< 45 degrees) and relatively large scale changes (factor of 10 resolution). These techniques have been generalized to the matching of semantic network descriptions generated by separate monocular analysis or from a baseline cartographic description [28].

Our interest in matching of monocular interpretations arises from our desire to relate structure descriptions generated from a building hypothesis system. The BABE Built-up Area Building Extraction system [4], performs monocular analysis on an image by extracting lines and corners and generating structure hypotheses. This work is similar to Huertas and Nevatia [13], but differs in that a large number of hypotheses are purposely generated such that buildings are rarely missed. These structures are then evaluated by a number of techniques such as shadow verification, shadow prediction, and shadow grouping [14]. The processes of verification, prediction, and grouping are used to rank order or prune the large number of BABE structure hypotheses.

3.1.1. Structure Verification

First of all monocular matching can be viewed as another form of structure verification. That is, sets of independently derived hypotheses from different images are matched using the scene registration model to relate absolute ground position in the two images. The results of this matching provide information that is not available in a single image, including an estimate of structure height and the reliability of each hypothesis. For example, because matching allows multiple hypotheses in one image to correspond with a single hypothesis in the secono image, we can use this fact to guide a re-examination of the structure delineation in the first image. The fragmentation of structures is a common source of error in computer vision, and understanding fragmentation requires some external process to predict its occurrence or to identify situations where it has occurred. Even in cases where there is a good one-to-one match between structures, different viewing angles, accidental alignments of objects in the scene, or differences in imaging conditions will produce differences in the segmentation which can be recognized as cues for further interpretation.

The use of high-level image cues such as aligned or oriented structures composed of lines, corners, and surfaces for perceptual grouping and stereo matching has recently seen some research activity [7, 23]. In our examples we have focused on the verification and grouping of hypotheses in order to improve monocular analysis. The determination of height is only one component of the matching process, rather than the primary result. We also used this matching as another way to select control points automatically and to perform scene registration.

3.1.2. A Matching Experiment

In this section we describe our matching results on a portion of a stereo pair of Los Angeles International Airport (LAX) used by Huertas and Nevatia in their building extraction research [13]. Figures 3-1 and 3-2 show the results of BABE hypothesis generation on the left and right image, respectively. The BABE results have been pruned automatically using shadow analysis [4]. As is evident in these results, BABE generates hypotheses for most of the buildings in the scene. However, it is apparent that there are differences in the quality of the delineation and in detection errors between the two images. In the ideal case, we should have the same number of building hypotheses in each image. Further, the roof delineation of each building should be quite similar but not identical, because of the displacement due to height.

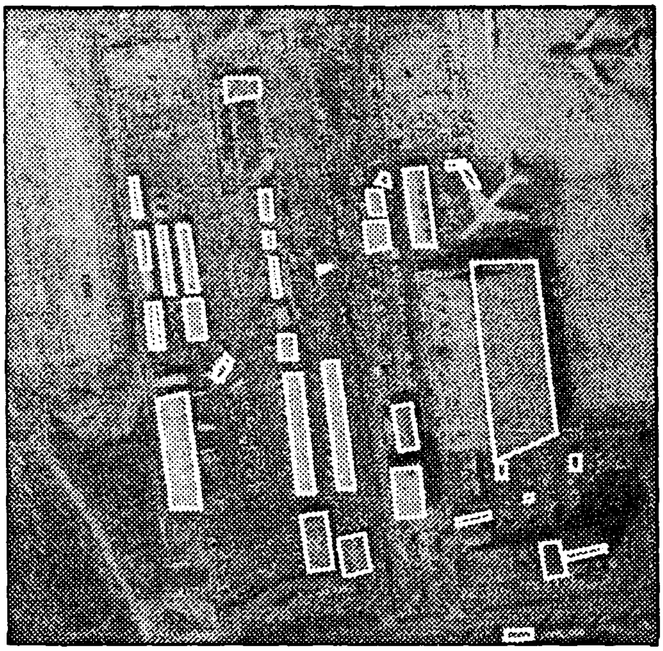


Figure 3-1: BABE building hypotheses for LAX [left image]

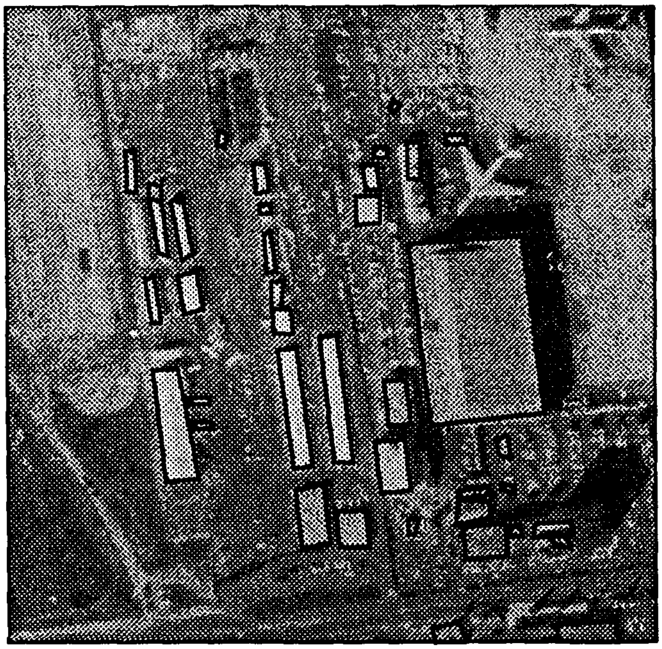


Figure 3-2: BABE building hypotheses for LAX [right image]

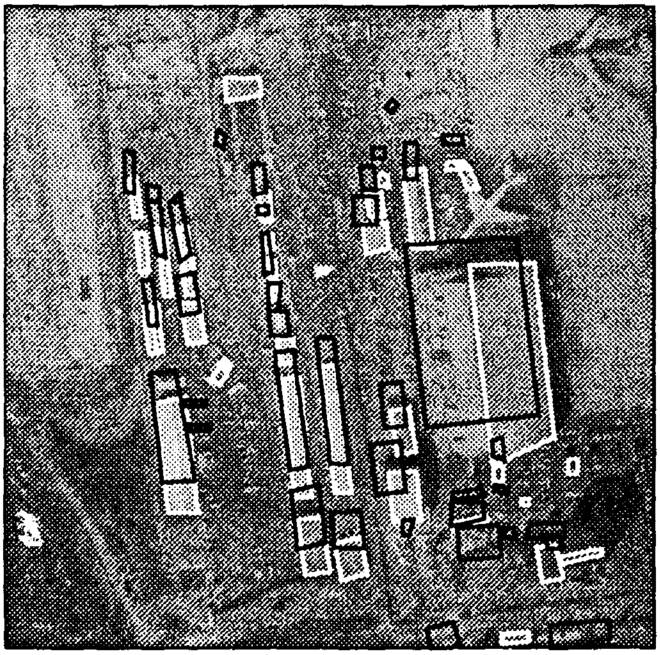


Figure 3-3: BABE results superimposed on the left image using coarse registration

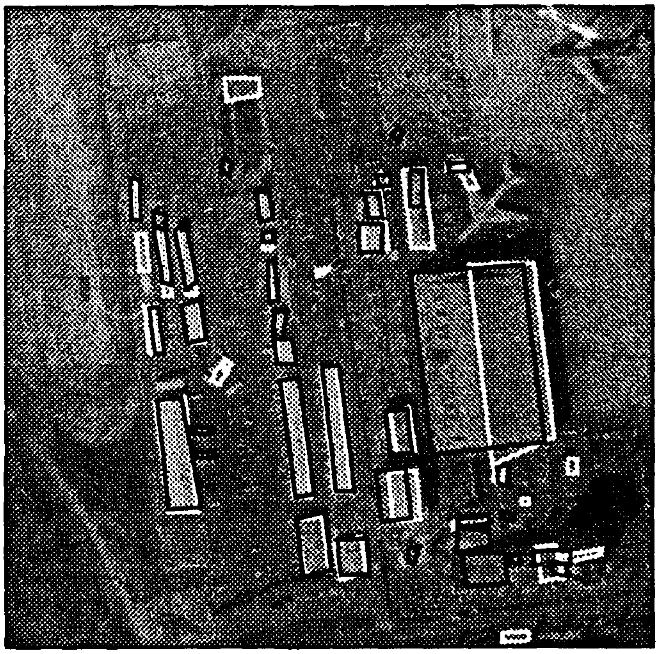


Figure 3-4: BABE results superimposec ɔn the left image using fine registration

Figure 3-4 shows the superimposition of BABE results on the LAX left image using the fine registration technique described in Section 2.4 while Figure 3-3 shows the accuracy of the coarse registration described in Section 2.3. This superimposition allows us to see directly the structure correspondence as well as the differences between the two monocular analysis results in Figures 3-1 and 3-2. The horizontal displacement between the hypotheses for the fine registration can be related to the relative height of the structures because of the epipolar constraint. Coincident structure hypotheses give very strong support for hypotheses of buildings. This is due to the fact that the feature extraction process rarely fails in exactly the same way in each of the images.

To automate matching between the hypotheses in the left and right images we utilize geometric constraints. We take each BABE hypothesis from the left image and find the best corresponding hypothesis on the right image. To evaluate the effects of the registration, we performed this matching on both the coarse registration using the CONCEPTMAP scene model, and the fine registration described above. Figure 3-5 shows the matching results using the CONCEPTMAP registration while Figure 3-6 shows the matching results of the fine scene registration. In both cases we have chosen a small area in the left-center of the LAX scene to illustrate the details of matching.

The matching process is a global search between two sets of boxes according to local limitations in the search area. The epipolar geometry of the fine registered images can be used to constrain the search area to a range of scanlines in the images. Then we use a simple criteria to select potential matches: the position of the hypothesis center-of-mass projected into a rectangular search space and amount of overlap between the pairwise structures.

This simple matching process allows us to consider arbitrarily complex polygonal structures because we are not performing discrete vertex or structure matching to establish a stereo correspondence such as in Mohan and Nevatia [23]. This is required given the relatively complex imagery and imprecise segmentation delineation provided by BABE. In many cases detailed high-level structure matching (as in [23]) will be defeated by errors in monocular feature detection due to occlusion, texture, and accidental alignments of objects and background. These are precisely the errors that cause area-based and feature-based matching to fail, although propagated to a high-level matching process.

Because our matching criteria are not very selective, we must disambiguate among many plausible matches. However, even if we devised a more specific set of match criteria, it is unclear whether we could account for situations in built-up urban areas where the buildings are very close, and have very similar shapes, orientations, and heights simply by using a set of local optimal matches. There are several examples of the alignment of similar buildings even within the LAX imagery. Thus, there is a virtue in the application of weak constraints because they do not require detailed high-level knowledge about the mis-registration. Instead of trying to disambiguate the matches locally we use global considerations based on the plausibility of the matched sets of structures. We define four different situations that occur depending on whether several structure hypotheses share the same correspondence with a hypothesis structure in the other image.

Tables 3-1 and 3-2 show the results of the monocular matching using the coarse and fine scene registration. In this experiment we search for the best match for each of the building hypotheses produced by BABE in the left image. A similar analysis could be performed on the structures generated from the right image. Four different situations can occur during matching that correspond to the application of local and global properties:

- **Type 1** This case corresponds to the "ideal" situation where we have a unique correspondence between a single hypothesis in the left and right image. The score of the correspondence gives us an estimate of the quality of the match between the two structures. A match score greater than 0.9 indicates that the two structures are quite similar. Pair (10,29) is an example of such a good match. A lower match score, as in the case of Pair (8,11), indicates that while there is a correspondence between structures their BABE delineation is not completely consistent between the left and right image.

- **Type 2** This case occurs when a structure in the left image shares a right structure correspondence with other structures in the left image. This correspondence is therefore ambiguous. In this case the match score is not sufficiently different to disambiguate between the multiple choices. However, knowledge of a reasonable height range for these structures could be used to select the correct correspondence. For example, Pair (28,12) and Pair (29,12) have a significant difference of 4.5 meters in their height estimate. However, neither height is sufficiently unusual to prefer one interpretation over another without some external information. However, in the case Pair (7,10) for the coarse registration, this match could be discounted due to a height interpretation that is below the local terrain.

- **Type 3** This case occurs when several matches were possible with different structures in the right image, for example Pair (26,3). The correspondence selected has the highest confidence match but other correspondences are possible, i.e., Pair (26,10). Once again, knowledge about the reasonable

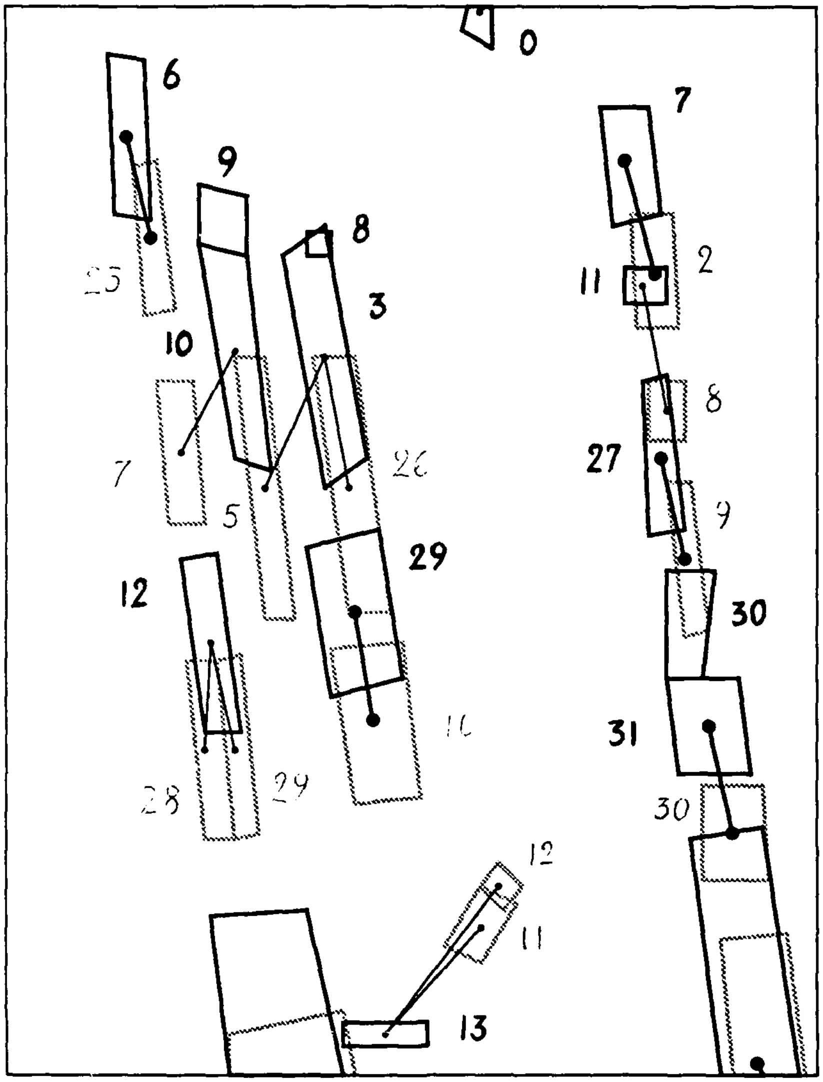


Figure 3-5: Matching of buildings using a coarse registration

Results of the matching of boxes for the coarse registration					
Type of corres.	Left box	Right box corres.	Corres. score	rel. height estimate	rel. line offset
1	2 8 9 10 25 30	7 11 27 29 6 31	0.92 0.73 0.85 0.94 0.83 0.93	4.0 3.5 4.4 3.7 3.8 4.2	18.2 21.0 16.4 18.3 17.4 18.2
2	7 28 29	10 12 12	0.73 0.89 0.80	-9.3 -0.1 4.4	16.9 17.5 17.1
3	5 26	3 3	0.75 0.87	-10.0 4.5	22.0 21.5
4	11 12	13 13	-44.5 -34.5	19.6 16.0	-24.9 -18.5

Table 3-1: Matching Results For LAX Building Hypotheses with coarse registration

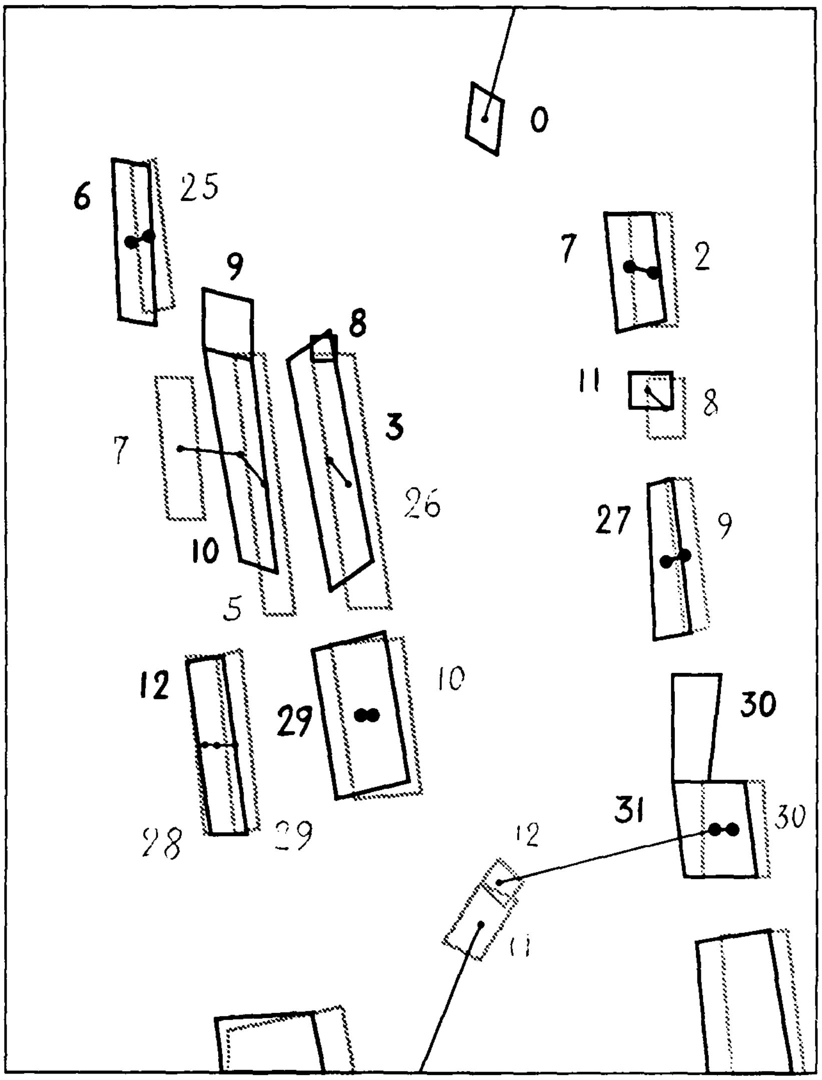


Figure 3-6: Matching of buildings using a fine registration

Results of the matching of boxes for the fine registration					
Type of corres.	Left box	Right box corres.	Corres. score	rel. height estimate	rel. line offset
1	2	7	0.92	3.1	0.2
	8	11	0.73	2.5	3.0
	9	27	0.86	3.4	-1.6
	10	29	0.94	2.7	0.3
	25	6	0.83	2.8	-0.6
	30	31	0.93	3.2	0.2
2	5	10	0.75	3.7	4.9
	28	12	0.89	-1.1	-0.5
	29	12	0.81	3.4	-0.9
3	26	3	0.87	3.5	3.5
4	7	10	-11.4	-10.3	-1.1
	11	31	-44.6	-35.5	-1.6
	12	13	-51.5	15.0	-36.5

Table 3-2: Matching Results For LAX Building Hypotheses with fine registration

height range for structures could be used to select the appropriate correspondence.

- **Type 4** This case occurs when structures in the left image do not have any reasonable correspondence in the right image such as in Pair (11,13). Nevertheless the best correspondence is given. The utility of such a match is to provide another analysis process with a context to search for new structures in the right image or to eliminate this structure from further analysis.

As we have seen, the ability to perform scene registration allows for the efficient correlation of monocular analysis. There appears to be no major difference between the matching results using the coarse or fine scene registration. In the case of good matches, i.e., Type 1, having a high score, the results are identical for the coarse and the fine registration. Even if, according to the registration, different types of matching are possible for the ambiguous matches, this technique still seems to be quite robust. However, different scene clues can be derived by the analysis of the absolute match score, the match type, the estimated height, and the relative row offset for the different registrations. For example, in Type 2, matching a choice between two competing interpretations must be made. A verification process could be invoked to locate a better matching and delineation of the structures. The estimate of structure height in the fine registration can be compared to the results of specific stereo matching algorithms. We can also characterize the quality of the scene registration (epipolar geometry in particular), by the relative row offset results for the matching of a given set of accurate structure delineations. For example, given a reference manual segmentation of buildings in the left and right images we can use the local translations in rows and columns between the matched structures in the coarse registration (i.e. the estimated height and the relative row offset) as estimates of the misregistration. Actually a simple analysis of the translations for the good matches in the coarse registration is used to set a spatial constraint to reduce the search window while refining the matching process. The relative row offset in the coarse registration is approximately 18 pixels. This can be derived by averaging the row offset of the good matches and can be verified by subtracting the relative row offset of the fine registration from that of the coarse registration.

Thus, the matching of high-level structures extracted from monocular analysis can be used for evaluation of the quality of the structure delineation, height estimation, or provide a refined estimate of the scene registration. In the following section we discuss in more detail the use of structure matching to refine the registration.

3.1.3. Structure Matching for Registration

As we have seen, starting from the coarse registration, we are able to perform structure matching that provides an estimate of the local offsets in rows and columns for the center-of-mass of the structures generated by BABE in the left and right images. We can consider these corresponding points as control points selected automatically and then perform a registration of the stereopair exactly as with the shadow corners.

This observation can be generalized, and we can consider the automated registration more globally. As we have seen, one requirement for automated registration is the automatic selection of corresponding points in the stereo pair images. Classically these points are physical features of the images like shadow corners, road intersections or specific marks, but we can also think of "virtual geometric points" defined by a geometrical relationship to real features in the images. The center-of-mass of the structures generated by BABE belong to this category.

With such a definition of control points, the automated registration can be performed using many different features such as isolated points, edges, boundary contour, regions, and structures. The problem is, as before, to perform accurate matching in order to end up with control points whose position is accurately known. Our ability to match various features is the characteristic that tends to limit our choice of techniques.

The traditional "characteristic points" approach is often accurate in point position, but because it is highly local in nature it is very difficult to find an accurate one-to-one correspondence. The more inherent complexity in the feature selected as a match point, the more constrained the matching becomes. For example, single intensity points in the images are very accurate in position but the matching process to find the correspondences is very difficult. On the other hand, very complex structures can be very easily disambiguated to find good correspondences. The problem is that as we match high-level structures our ability to determine match point positions becomes less accurate. Because complex features such as road intersections, building outlines, unique terrain points require a detailed analysis, their detection and delineation may be inexact. From these inexact corresponding structures we must generate "virtual" control points whose position in the feature is less sensitive to these errors.

The Figure 3-7 shows the "virtual" control points we selected in the images using the matching of structures generated by BABE. The structures generated by BABE in the left image are shown in white, the right image structures are shown in black. The automatic control points selected correspond to the good structure matches. Given that BABE does a fairly good job in structure delineation and generates consistent hypotheses in both images, the set of control points considered is quite reliable. In fact, the residuals of the registration shown in Table 2-2 are comparable to the registrations using a manual set of control points.

Figure 3-8 represents the BABE results superimposed on the left image using the isometric registration derived using these control points. The results are nearly identical to those derived using manual registration and show a good superimposition of the building structures.

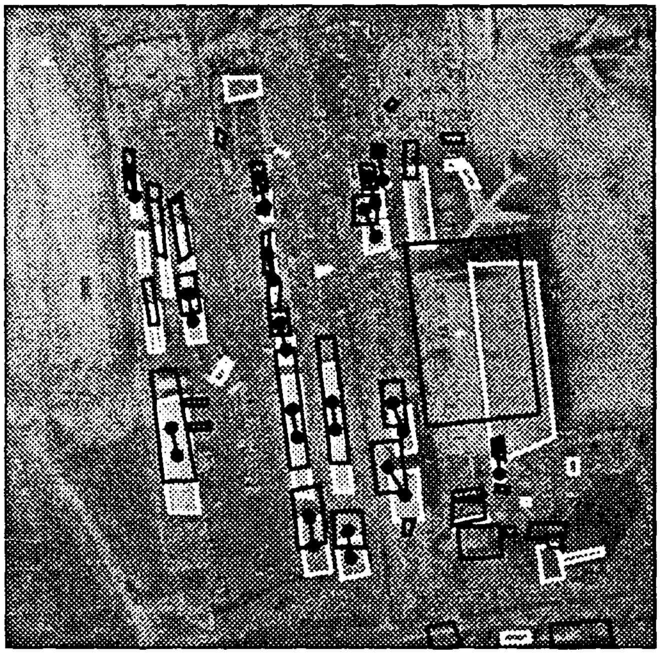


Figure 3-7: Structure matching using coarse scene registration

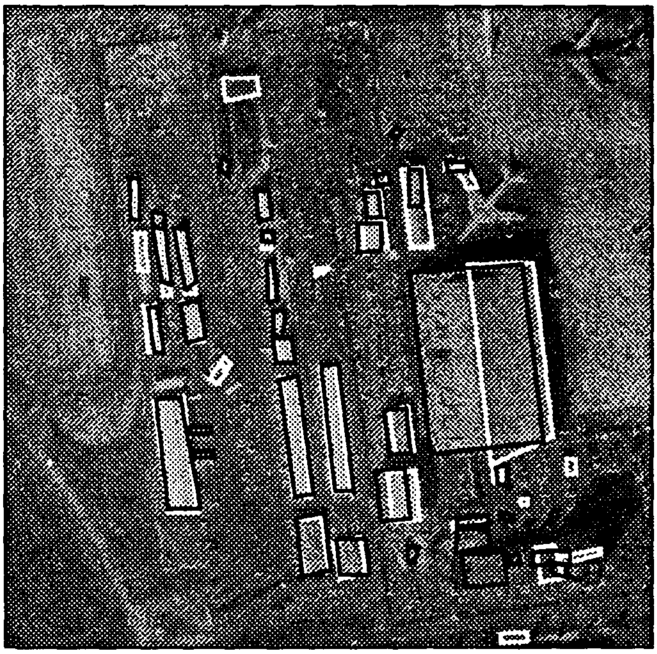


Figure 3-8: Superposition of structures using structure matching registration

3.2. Registration for stereo matching

The central issue in computational stereo is the solution of the correspondence between features visible in two overlapping images. The correspondence of a point feature visible in the left and right image of a stereo pair can be used to generate the three-dimensional description of that point in the scene. Points need not be the only feature matched. As we have seen, the result of matching structures generated by monocular analysis can be considered as stereo matching and yields a relative height estimate.

In Section 2.2 we discussed how the epipolar constraint is used to simplify stereo matching by reducing it to a one-dimensional problem. This is because the epipolar lines in the imagery are registered to be corresponding scanlines in the left and right image. The assumption that the scene registration is ideal and that the epipolar constraint is totally satisfied is rarely warranted in imagery digitized from aerial photography. This is due to all of the orientation problems previously discussed as well as inaccuracies inherent to the digitization process primarily due to rotation of the image. In the following section we discuss the effect of coarse and fine scene registration on two stereo matching algorithms.

3.2.1. Two stereo correspondence algorithms

Algorithms for stereo correspondence can be grouped into two major categories: area-based and feature-based matching. Examples of area-based matching include correlation techniques for matching image intensity patches using various evaluation functions including normalized cross-correlation, mean-square-difference, or surface fitting residual error. Feature-based techniques match image features derived from edge, line, or boundary detection. Area-based techniques provide a dense disparity map with an estimate generated at every point in the image. Feature-based approaches provide depth information only at points where the features are generated, often points of intensity discontinuity that may correspond to discontinuities in depth.

Both classes of techniques, area-based and feature-based, have advantages and drawbacks that primarily depend on the task domain and the three-dimensional accuracy required. For complex urban scenes, feature-based techniques appear to provide more accurate information in terms of locating depth discontinuities and in estimating height. However, area-based approaches tend to be more robust in scenes containing a mix of buildings and open terrain. For this reason we have developed two stereo matching algorithms. S1 is an area-based algorithm and uses the method of differences matching technique developed by Lucas [15, 20]. S2 is feature-based using a scanline matching method that treats each epipolar scanline as an intensity waveform. The technique matches peaks and troughs in the left and right waveform. Both are hierarchical and use a coarse-to-fine matching approach. Each is quite general as the only constraint imposed is the order constraint for the feature-based approach. The order constraint should generally be satisfied in our aerial imagery except in cases of hollowed structures.

Both matching algorithms assume the epipolar geometry but have different sensitivity to its accuracy. The S1 area-based approach uses a hierarchical set of reduced resolution images to perform a coarse-to-fine matching of small windows in the two images. At each level the size of the windows for the matching process depends on the resolution of the reduced image. An initial disparity map is generated at the first level. Subsequent matching results computed at successively finer levels of detail are used to refine the disparity estimate at each level. Therefore the amount of error in the scene registration that can be tolerated by this matching algorithm depends on the size of the matching windows. However, since there is a relationship between the matching window size and the level of accuracy, simply using larger matching windows may not be desirable.

The S2 feature-based approach matches epipolar lines in the left and right image. It uses a hierarchical approximation of the intensity waveforms to match peaks and valleys at different levels of resolution. To avoid mismatches it uses inter-scanline consistency that enforces a linear ordering of matches without order reversals. It also applies an intra-scanline consistency that considers the matches in adjacent scanlines. Application of intra-scanline constraint is used to increase the confidence of matches found to be consistent across multiple scanlines and to delete improbable matches.

Figure 3-9 is a complex industrial area scene and was the focus of our discussion on coarse and fine scene registration in Sections 2.3 and 2.4. This scene contains many of the difficulties found in stereo matching, including occlusion, complicated textures, large depth discontinuities, and complicated three-dimensional objects. The Figure 3-10 shows the results of the matching for the CONCEPTMAP coarse registration using the area-based algorithm S1. In all of the disparity match results presented in this paper, brighter regions are closer to the camera and have greater height. Darker regions are at or below the relative terrain ground plane. The results using the coarse registration are quite errorful. We can barely discern the general shape of the taller buildings in the middle and the upper left areas of the scene. The S2 algorithm is completely unable to use a coarse registration since the scanline matching assumes that the epipolar constraint is satisfied.

The Figure 3-11 shows the results of the matching using the S1 algorithm with the fine registration produced by the manual selection of shadow corner points. The matching results are significantly better with the bright areas again representing the highest regions and corresponding to most of the buildings in the scene. Although the delineation is not crisp there are few major mismatches, and we have an adequate impression of the range of heights in the scene. The S1 algorithm has many of the advantages and drawbacks of any area-based technique. As we can see in Figure 3-11 most of the errors are due to abrupt changes in height due to man-made structures.

The Figure 3-12 shows the results of the matching using the S2 algorithm with the same fine scene registration in Figure 3-11. This technique performs very well on the disparity discontinuities caused by man-made structures, and therefore we have a much better delineation of the buildings than in Figure 3-11. Nevertheless, despite post processing of the disparity results, the resulting disparity image is noisy. As expected, the S2 algorithm can not provide robust matches in areas of uniform intensity or in highly textured areas.

The results of the two stereo matching algorithms are quite complimentary, and we believe that it is possible to take advantage of the different failure modalities in order to form a composite disparity map that gives a more accurate three-dimensional representation of the scene. It is also clear that stereo matching relies on a more accurate scene registration than is provided by the coarse registration described in Section 2.3. Even when the matching window size for an area-based stereo algorithm is larger than the inherent mis-registration, it may be difficult for the matching algorithm to recover from mis-matches due to poor scene registration. This is in contrast to the results in structure matching presented in Section 3.1 that appear to be much less sensitive to a coarse scene registration.

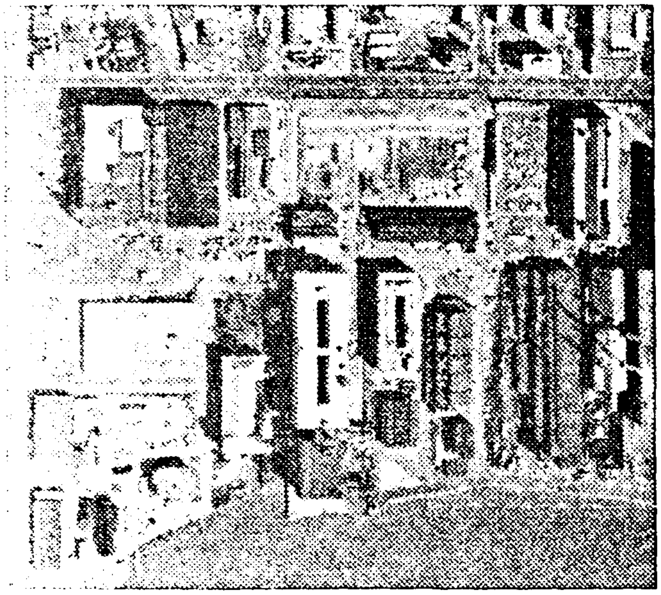


Figure 3-9: Intensity image of the scene

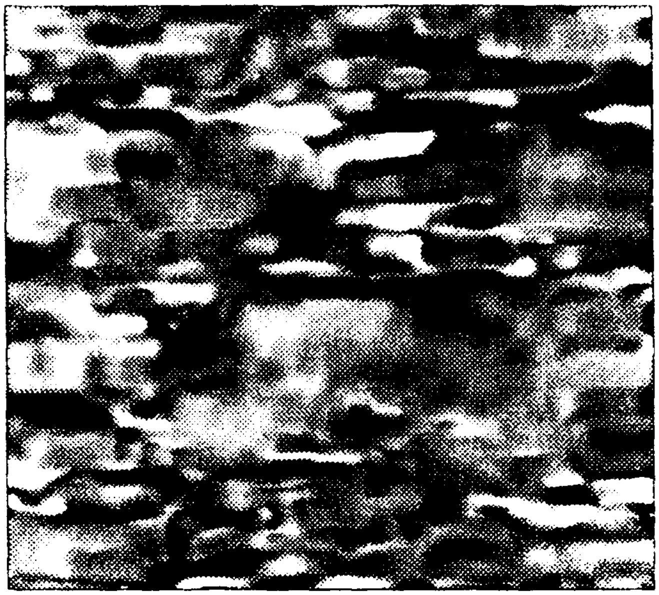


Figure 3-10: S1 disparity map using coarse registration

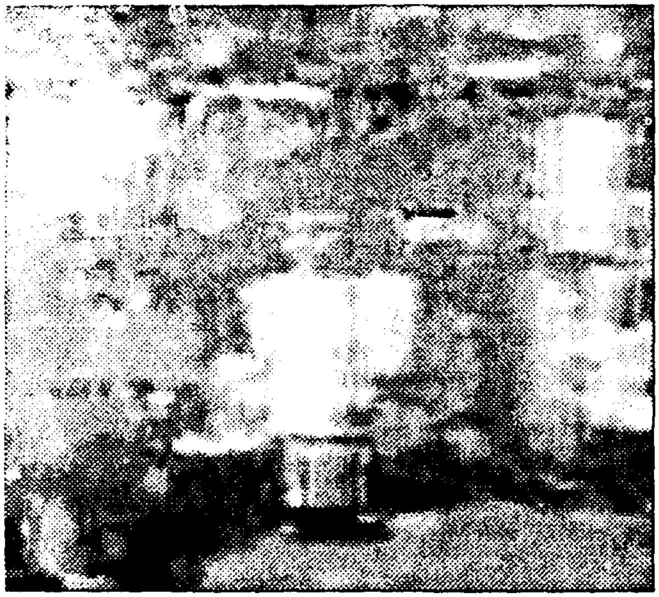


Figure 3-11: S1 disparity map using fine registration

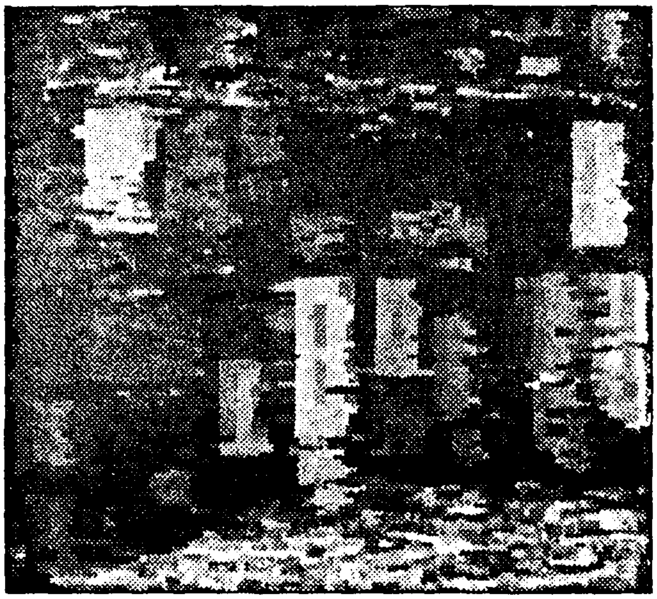


Figure 3-12: S2 disparity map using fine registration

[illegible] Registration for visualization

[illegible] is needed in order to understand the quality of the stereo [illegible] of a left and right stereo image from the original unregistered [illegible] the scene as the human eye will naturally [illegible] simultaneously [illegible] constraint [illegible]

[illegible]

[illegible] glasses to separate the left and right images. Several manufactures use [illegible] the two stereo fields alternating at 1/60th or 1/120th of a second. The detail and [illegible] is much crisper and easier for viewers to fuse than using the anaglyph technique. [illegible] we have used the anaglyph method; however the construction of a synthetic stereo image is [illegible] regardless of viewing method.

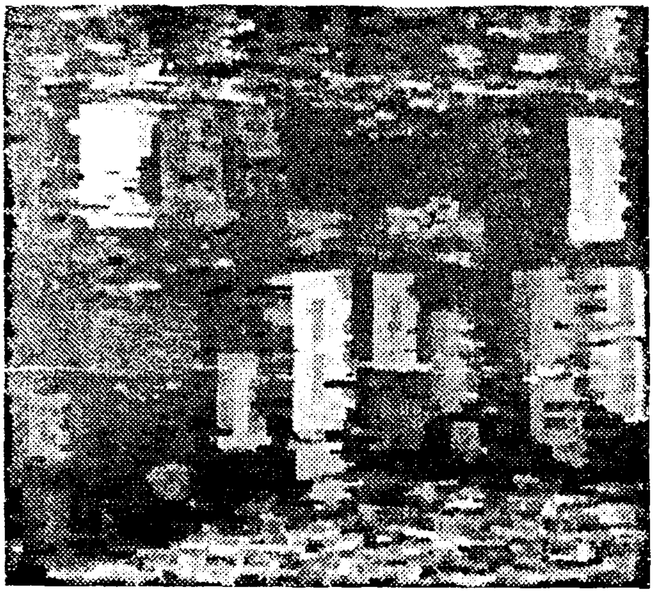


Figure 3-13: S2 stereo matching result [left]

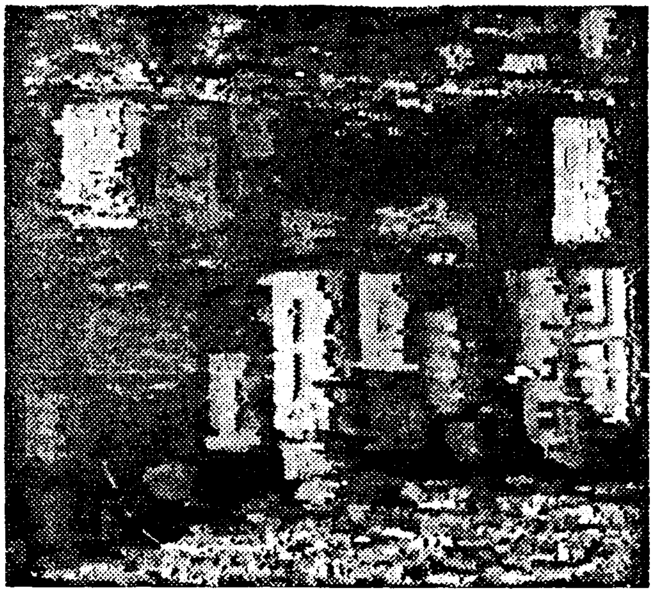


Figure 3-14: *S2 stereo matching result [right]*

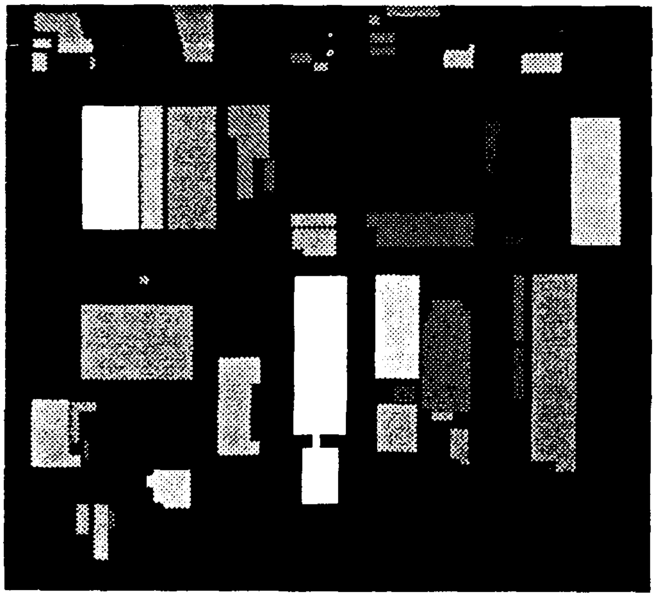


Figure 3-15: Ground truth left image for DC38008 scene

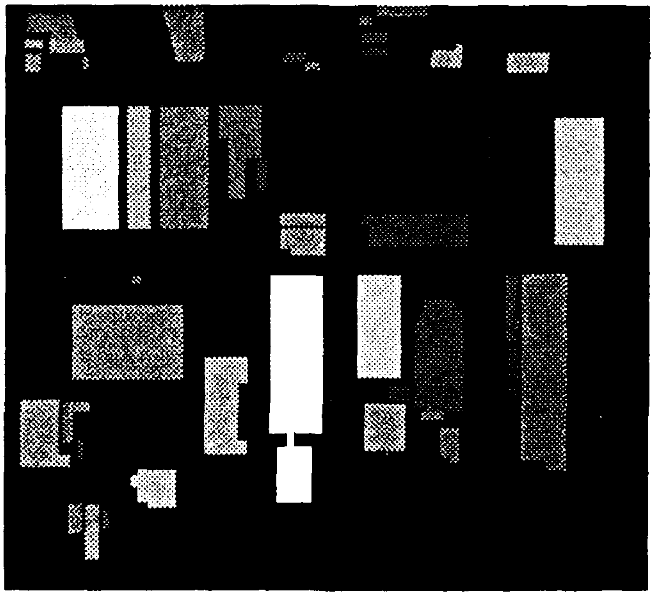


Figure 3-16: Ground truth right image for DC38008 scene

3.3.1 Construction of the right image from the left image

Given a left and right image registered into the epipolar geometry, as are the intensity images after fine registration, we can display them using different stereo viewing methods as described above. This is the most common way to achieve stereo visualization. However, a more interesting approach is to use this representation to generate a stereo pair (left and right image) from the three dimensional information we have computed by stereo matching in order to directly visualize the matching results as a three-dimensional scene. We call the generation of a stereo pair from three dimensional information derived from stereo analysis *synthetic stereo* *reconstruction*. Synthetic reconstruction can be used to visualize and compare the results of stereo matching by direct visualization.

A relative height computation, or disparity, is the result of most stereo matching algorithms. The disparity is usually encoded in a representation related to the geometry of the left image. In this disparity map, the values of each point in the map correspond to the relative height of that point in the left image. In order to generate a synthetic reconstruction containing the information extracted by the matching process we must generate a new right image. Each point in the right image is the corresponding point in the left image displaced by the relative height estimate in the disparity map. This process is exactly the opposite of that used to generate the disparity map. However, we do not have to solve the matching problem. The computed right image is, by definition, perfectly registered since there are only local horizontal shifts between the left and the right image. Thus, we satisfy the epipolar constraint. Figure 3-14 and Figure 3-13 shows reconstructed synthetic stereo images for the stereo matching results produced by S2 in Figure 3-12.

3.3.2. 3D segmentation for ground truth determination

The visualization of scenes and stereo matching results is a powerful tool for the qualitative comparison of different scene interpretation techniques. One technique is the side-by-side comparison of the original stereo scene and the automated reconstruction. Such a comparison allows us to quickly see those buildings that are missing or have errors in height or ground position.

However, a quantitative approach is also possible and is potentially more useful. Using the anaglyph display techniques we can generate a three-dimensional segmentation that allows us to store the structure of each building in the scene. The form of the data is simply a segmentation description file containing collections of left image points and their relative height. From that representation we can infer a partial three-dimensional representation of the buildings guessing the shape of the invisible parts, much as is done with simple wireframe models. We can also use this representation as a baseline reference representation for buildings in order to compare and contrast the various processing results.

Figures 3-15 and 3-16 show how this technique can be used to construct a simple three-dimensional ground truth segmentation that can be visualized as a stereo scene. We simply represent the building roofs as horizontal surfaces and displace these surfaces proportionally to the actual height of the buildings that we got from the three-dimensional segmentation of the roofs. Such a stereo pair is very useful in order to compare the results of automatic scene matching.

4. Automatic scene registration

As we have seen, structure matching, stereo matching, and visualization all rely on the quality of the stereo registration. The registration and the matching process are therefore interdependent. Structure matching appears to be a task that we can reliably perform even with a coarse scene registration. Further, the results of structure matching provides a method to automatically refine the initial coarse scene registration. In this Section we demonstrate a complete end-to-end scenario of automatic structure matching, fine registration, and stereo analysis. Thus, we can automatically generate a three-dimensional representation of the scene starting from the CONCEPTMAP image database.

We began with the DC38008 test area corresponding to Figures 2-3 and 2-4, previously shown in Section 2.3. We then utilize the BABE structure results and perform structure matching to select reliable control points. The structures generated by BABE are fragmented and are not as consistent as those generated for the LAX stereo pair. Nevertheless, we are able to find a number of good matches, well distributed across the image, as shown in Figure 4-1. Subjectively, the registration quality is good, as seen in Figure 4-2, where many of the building fragments are now aligned. The overall registration quality is detailed in Table 2-1 (ISO structure) in Section 2.4. While it is not as accurate the registration derived by manual ground control selection, it is clearly comparable.

Finally using this automatically registered stereo pair we performed stereo matching to get a dense disparity map of the scene. Figures 4-3 and 4-4 show the results for the S1 and the S2 matchers. The results are comparable to those in Figures 3-11 and 3-12 achieved using manual selection of control points. Thus we have shown the feasibility of end-to-end processing to establish precise local registration using automatic ground control point estimation.

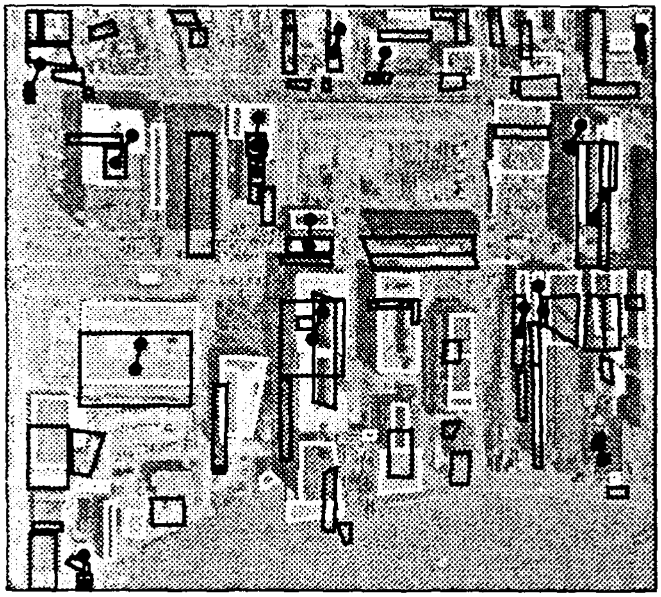


Figure 4-1: Automatic control points using structure matching

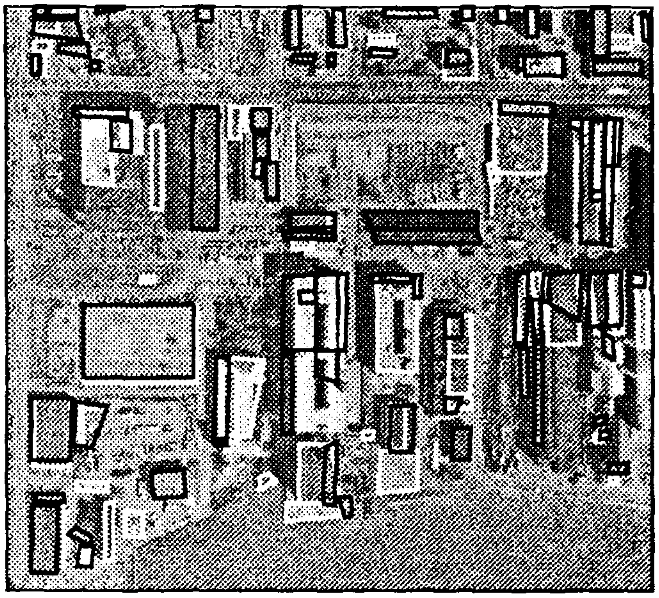


Figure 4-2: Superposition of structures using structure matching registration

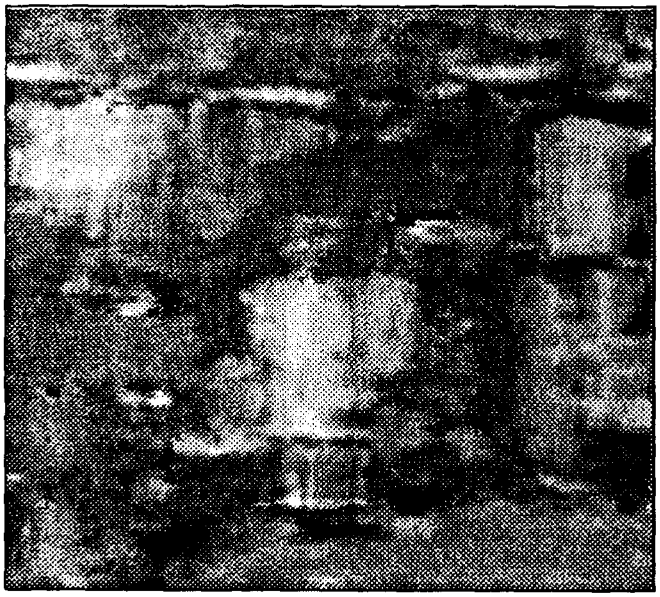


Figure 4-3: S1 disparity map using structure matching registration

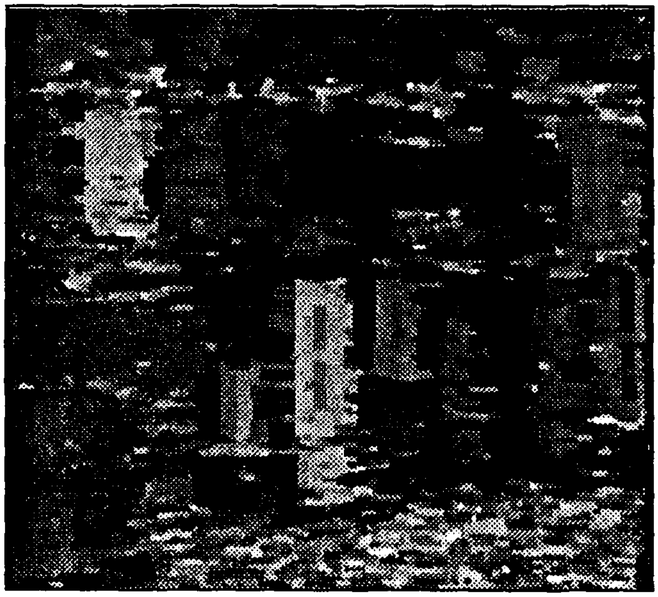


Figure 4-4: S2 disparity map using structure matching registration

5. Conclusions

The importance of scene registration in the automated interpretation of aerial imagery can not be overstated. Scene registration is required for monocular matching, stereo analysis, scene visualization, accurate mensuration, and for many other photo-interpretation tasks. Most work in computational stereo has ignored the problem of scene registration assuming that the left/right image pairs were already in epipolar geometry. As we have seen, this may limit the utility of many feature-based and some area-based matching techniques, especially in cases where there are significant residual errors in the registration process.

Traditionally we have separated the stereo analysis of digital images in two problems, registration and matching, and have attempted to solve each independently. However, the results of matching, whether structural or using a stereo model, are actually the ultimate form of scene registration since the matching solves the correspondence between different objects in the images. In some sense registration and matching are corresponding processes that are performed at different representational levels. Registration relies on mathematically modeling the image acquisition in the three-dimensional world, while matching defines the relationship between corresponding points in two images. Registration is generally necessary to constrain search during matching, but at least a sparse matching is necessary to perform the registration.

5.1. Future Work

There are several areas for future work focused on improving techniques for scene registration. First, using the image-to-map correspondence, we need to improve the accuracy of the landmarks stored in the CONCEPTMAP database. It may be necessary to include accurately surveyed geodetic control points in addition to those that are acquired from map sheets. However, given the high resolution of the imagery that we are working with, it may actually be quite difficult, using these control points, to improve on the current level of accuracy. More complex known landmarks such as road networks may be utilized to accurately register imagery to maps for automated scene registration [16].

For direct image-to-image correspondence we have seen some limitations in automatic extraction of shadow corner points in complex urban imagery for registration. Additional sources of reliable registration points should be available using monocular extraction of man-made structures such as the road networks. From our previous work in road detection [3] and tracking [19] it seems quite reasonable to use these structures as potential landmarks for scene matching. Furthermore the model of transformation between the two images should be enhanced to get a better registration once we have accurate control points.

Finally, we need to pursue the experiment of iterative refinement of registration via coarse-to-fine matching. The basic idea is to perform a coarse registration using road networks, or known ephemeris data. This coarse registration would be followed by feature matching using structure matching or shadow corners to generate an estimate of the error in the registration. This estimate can be refined using a hierarchical approach using high-level features that are easily matched but have inaccuracies in position, to low level features that are difficult to match but have unambiguous positions in the stereo pair. The process of registration and matching would then iteratively converge to a complete matching of the scene. Whether this approach can achieve high registration accuracy equal to manual correspondence is a topic for further research.

6. Acknowledgements

We thank our colleagues in the Digital Mapping Laboratory for an interesting and congenial working environment. We particularly acknowledge the contribution of Yuan Hsieh and Aviad Zlotnick to the development of several of the examples used in this paper.

7. References

[1] R.D. Arnold.
Local Context in Matching Edges for Stereo vision.
In *Proceedings: DARPA Image Understanding Workshop*, pages 777-791. May, 1978.

[2] Aviad, Z.
Locating Corners in Noisy Curves by Delineating Imperfect Sequences.
Technical Report CMU-CS-88-199, Carnegie-Mellon University, December, 1988.

[3] Aviad, Z. and P. D. Carnine.
Road Finding for Road Network Extraction.
In *Proceedings: Computer Vision and Pattern Recognition*, pages 814-819. Ann Arbor, Michigan, June, 1988.
Extended version available as Tech. Report CMU-CS-88-157.

[4] Aviad, Z., McKeown, D. M., Hsieh, Y.
The Generation of Building Hypotheses From Monocular Views.
Technical Report, Carnegie-Mellon University, January, 1989.
to appear.

[5] S. T. Bamard.
Stochastic stereo matching over scale.
In *Proceedings: DARPA Image Understanding Workshop*, pages 769-778. April, 1988.

[6] S.T. Barnard and M. A. Fischler.
Computational stereo from an IU perspective.
In *Proceedings: DARPA Image Understanding Workshop*, pages 157-167. April, 1981.

[7] H. S. Lim and T. O. Binford.
Structural correspondence in stereo vision.
In *Proceedings: DARPA Image Understanding Workshop*, pages 794-808. April, 1988.

[8] R. A. Brooks, A. M. Flynn, and T. Marill.
Self calibration of motion and stereo vision for mobile robot navigation.
In *Proceedings: DARPA Image Understanding Workshop*, pages 398-410. April, 1988.

[9] L-H Chen, and T. E. Boult.
An integrated approach to stereo matching, surface reconstruction and depth segmentation using consistent smoothness assumptions.
In *Proceedings: DARPA Image Understanding Workshop*, pages 166-176. April, 1988.

[10] Faugeras, O. D., and Toscani, G.
The Calibration Problem for Stereo.
In *Proceedings of Computer Vision and Pattern Recognition*, pages 15-20. June, 1986.

[11] M. J. Hannah.
SRI's Baseline Stereo System.
In *Proceedings: DARPA Image Understanding Workshop*. December, 1985.

[12] B. K.P. Horn.
Relative Orientation.
In *Proceedings: DARPA Image Understanding Workshop*, pages 826-837. April, 1988.

[13] Huertas, A. and Nevatia, R.
Detecting Buildings in Aerial Images.
Computer Vision, Graphics, and Image Processing 41:131-152, April, 1988.

[14] Irvin, R. B., McKeown, D.M.
Methods for Exploiting the Relationship between Buildings and their Shadow in Aerial Imagery.
In *SPIE Proceedings Image Understanding and the Man-Machine Interface II*. January, 1989.
Also available as CMU Computer Science Technical Report CMU-CS-88-200.

[15] B. D. Lucas.
Generalized Image Matching By The Method of Differences.
PhD thesis, Carnegie Mellon University, July, 1984.

[16] Chris McGlone.
Automated image-map registration using active contour models and photogrammetric techniques.
In *SPIE Proceedings on Reconnaissance, Astronomy, Remote Sensing, and Photogrammetry.* January, 1989.

[17] McKeown, D.M.,
Digital Cartography and Photo Interpretation from a Database Viewpoint.
In Gargarin, G. and Golembe, E. (editor), *New Applications of Databases*, pages 19-42. Academic Press, New York, N. Y., 1984.

[18] McKeown, D.M.
The Role of Artificial Intelligence in the Integration of Remotely Sensed Data with Geographic Information Systems.
IEEE Transactions on Geoscience and Remote Sensing GE-25(3):330-348, May, 1987.
Also available as Technical Report CMU-CS-86-174.

[19] McKeown, D.M. and Denlinger, J. L.
Cooperative Methods for Road Tracking in Aerial Imagery.
In *Proceedings IEEE Computer Vision and Pattern Recognition Conference*, pages 662-672. June, 1988.

[20] D. M. McKeown, C. A. McVay, and B. D. Lucas.
Stereo Verification in Aerial Image Analysis.
In *Proceedings: DARPA Image Understanding Workshop*, pages 310-326. December, 1985.

[21] H. P. Moravec.
Obstacle Avoidance and Navigation in the Real World by a Seeing Robot Rover.
PhD thesis, Stanford University, September, 1980.

[22] N. M. Nasrabadi, Y. Liu, and J-L. Chiang.
Stereo vision correspondence using a multi-channel graph matching technique.
In *IEEE international Conference on Robotics and Automation.* April, 1988.

[23] R. Mohan and R. Nevatia.
Perceptual grouping for the detection and description of structures in aerial images.
In *Proceedings: DARPA Image Understanding Workshop*, pages 512-526. April, 1988.

[24] Y. Ohta and T. Kanade.
Stereo by Intra- and Inter-scanline Search using Dynamic Programming.
IEEE Transactions PAMI-7(2):139-154, March, 1985.

[25] *The Manual of Photogrammetry*
Fourth Edition edition, American Society of Photogrammetry, Falls Church, VA., 1980.

[26] Price, K. E.
Change detection and Analysis in Multi-spectral Images.
PhD thesis, Carnegie-Mellon University, December, 1976.

[27] Price, K. and Reddy, D. R.
Matching Segments of Images.
IEEE Transactions on Pattern Analysis and Machine Intelligence 1(1):110-116, January, 1979.

[28] K. E. Price.
Relaxation Matching Techniques_A Comparison.
IEEE Transactions PAMI-7(5):617-623, September, 1985.

[29] D. Weinshall.
Qualitative vs. quantitative depth and shape from stereo.
In *Proceedings: DARPA Image Understanding Workshop*, pages 779-785. April, 1988.

Lockheed Imaging Technology Research for Missiles and Space

Carolyn Bjorklund
Mark Noga

LMSC/R&DD/DIPL
3251 Hanovcr St.
O/96-30; B/251
Palo Alto, CA 94034

Eamon Barrett

LMSC/SSD/IPL
1111 Lockheed Way
O/61-40; B/107
Sunnyvale, CA 94089-3504

Darwin Kuan

LMSC/R&DD/AIC
3251 Hanover St.
O/96-20; B/259
Palo Alto, CA 94034

ABSTRACT

This paper summarizes various projects in the Lockheed Missiles and Space Company Image Technology Development Program. These include: knowledge-based systems, applications of neural networks, perspective invariant object recognition, space astronomy, materials research, integrated electronics and architectures, advanced software, image processing, automation and robotics. Image Understanding aspects of the projects are highlighted, with an emphasis on three-dimensional data derivation and visualization.

1. INTRODUCTION

This presentation has two objectives. First, we give a brief overview of Lockheed Missiles and Space Company's (LMSC's) activities in digital image processing and interpretation, highlighting aspects of the work that we have judged to be of interest to the present audience of experts in the image understanding community. Second, we present technical depth in one specific area, the utilization of three-dimensional information in imagery exploitation, since this subject plays a vital role in several of the DARPA Strategic Computing (SC) Vision application areas and programs, such as object modeling for photointerpretation, automated manufacturing and robotic manipulation.

Lockheed is a ten-billion-dollar corporation with major activities in the United States and throughout the world, hence the present paper cannot summarize all imaging research activities. We will focus on image algorithms and techniques developed at LMSC's Palo Alto and Sunnyvale laboratories, a major portion of Lockheed's image technology program. In common with a number of the major aerospace corporations, Lockheed develops a proportion of its imaging technology for controlled access programs, and thus this information must be sequestered. However, the authors feel confident that this latter restriction, while cutting down on specific details, will not detract from the accuracy of the overall picture.

At LMSC, image technology development is driven by the requirements of the company's various missiles and space programs. These requirements include:

- AUTOMATED SCREENING of visible, radar and sonar imagery from spacecraft, aircraft and oceangoing platforms;
- AUTOMATED INSPECTION of missile and aircraft parts and electronic components using various sensing technologies;
- AUTOMATED ASSESSMENT of imagery-derived features, with applications to collection system management and mission planning;
- IMAGE ANALYSIS WORKSTATIONS that facilitate interactive imagery exploitation and incorporate automated database access, analyst aids, and report generation capabilities.

A number of LMSC technology developers and managers, recognizing the high payoff that would result from successful automated image understanding systems in these areas, are optimistic about the migration path from the community's present comparatively modest accomplishments to significant capabilities for automated systems within a decade. Current estimates of the probable dollar value of image-related sales over the next five years have helped to assure that LMSC management will continue to view image technology development as an important component of the company's strategy for missiles and space business.

We now proceed with Part 2 of our presentation, an overview of LMSC image technology research. Part 3 treats three-dimensional issues in greater technical depth. Technical articles and video tape presentations on LMSC's work in several of these areas can be made available upon request.

2. OVERVIEW

In the following pages we summarize various major projects within LMSC. They include automated image screening and processing, knowledge-based technology, or rapid (video-rate) cue extraction and scene generation.

2.1 KNOWLEDGE-BASED SYSTEMS

2.1.1 AES

The Automated Exploitation System (AES) addresses requirements for monitoring objects of interest and performing situation assessment over large geographic areas. Figure 1 illustrates the four main subsystems of the overall architecture: Cue Extractor (CE), Knowledge-Based Exploitation (KBE), Interactive WorkStation (IWS) and a database subsystem. AES is partitioned in a manner intended to optimize the use of numerical and symbolic processes, algorithmic and heuristic components, and appropriate conventional and knowledge-based technologies. The CE processes raw image data, and identifies objects and target cues based on pixel and object-model data. Hence, the CE consists primarily of numerically intensive processes based on image texture, variance measurements, and pattern matching algorithms. Cues and image registration coefficients are passed to the KBE subsystem for screening and verification, situation assessment, and planning. This system combines the cues with ground-truth and doctrinal knowledge to determine their importance and generates reports on the screening results, plus evaluation of the cues. This information is passed to the IWS where an image analyst can monitor, observe, and evaluate system functionality as well as respond to critical items. The database subsystem stores and shares reference imagery, collateral information and digital terrain data to support both automated and interactive processing. This partitioning of functionality into subsystems facilitates hierar-

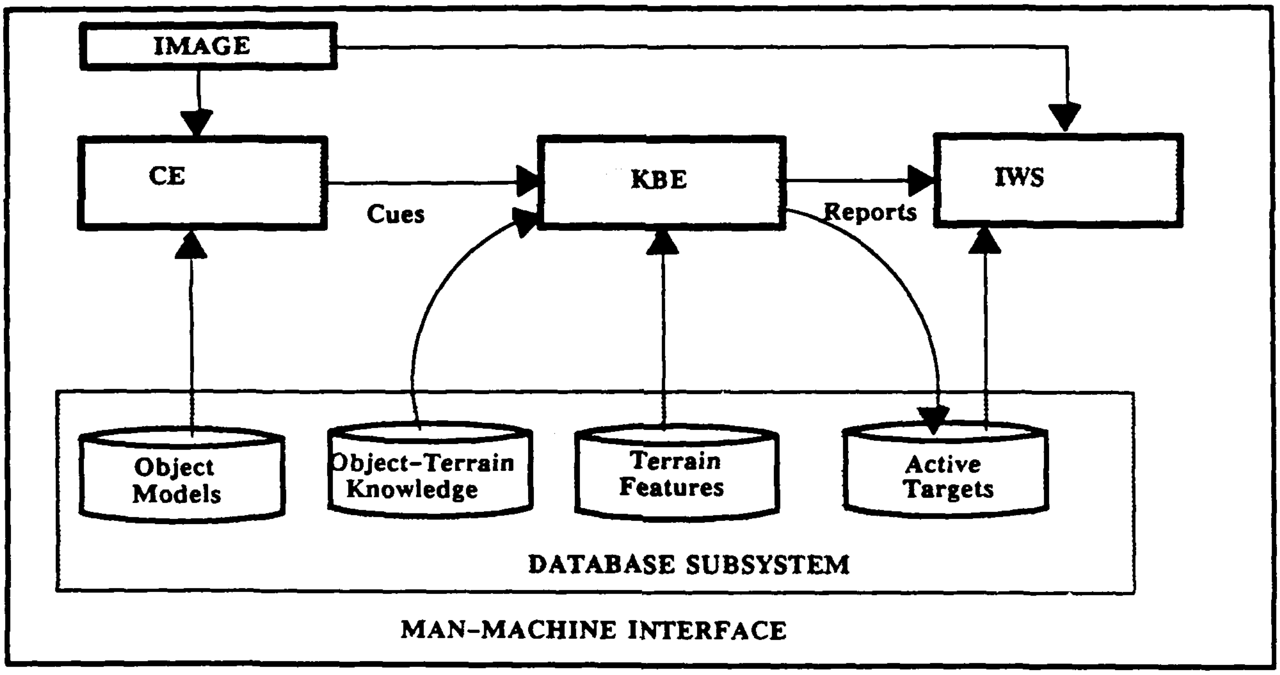


Figure 1. AES architecture.

chical application of knowledge in image interpretation. The current AES prototype helps in identification, capture, representation, and refinement of knowledge. The KBE subsystem software is written mainly in LISP, although the ART expert system shell is used in the implementation. The hardware testbed includes a Symbolics 3675 computer.

Initially, the CE performs analysis on raw imagery data and produces a list of vectors, (x,y) pixel locations, and notations as to the "likeness" of the selected positions in the image versus a specific catalog of objects (roads, rivers, and other cultural data) and targets (tanks, jeeps, and strategic objects). This cue-level information, along with rudimentary registration coefficients which transform the (x,y) locations into (latitude, longitude) space coordinates are provided to the KBE software.

The KBE accepts a list of these cues from the CE and uses various types of knowledge about terrain, object-terrain positioning doctrine, formation of object grouping, and historical data to screen the cues, then validates and assigns a status to each target. Finally, it outputs symbolic target descriptors to the IWS. The purpose of the KBE is to analyze and evaluate the outputs of the cue extractor and to provide significant cues to the image analyst to assist him or her in performing routine commonplace tasks more effectively. It accomplishes the former by providing figures-of-merit and expedites the latter through maintenance of database information and interpretation of target cue data.

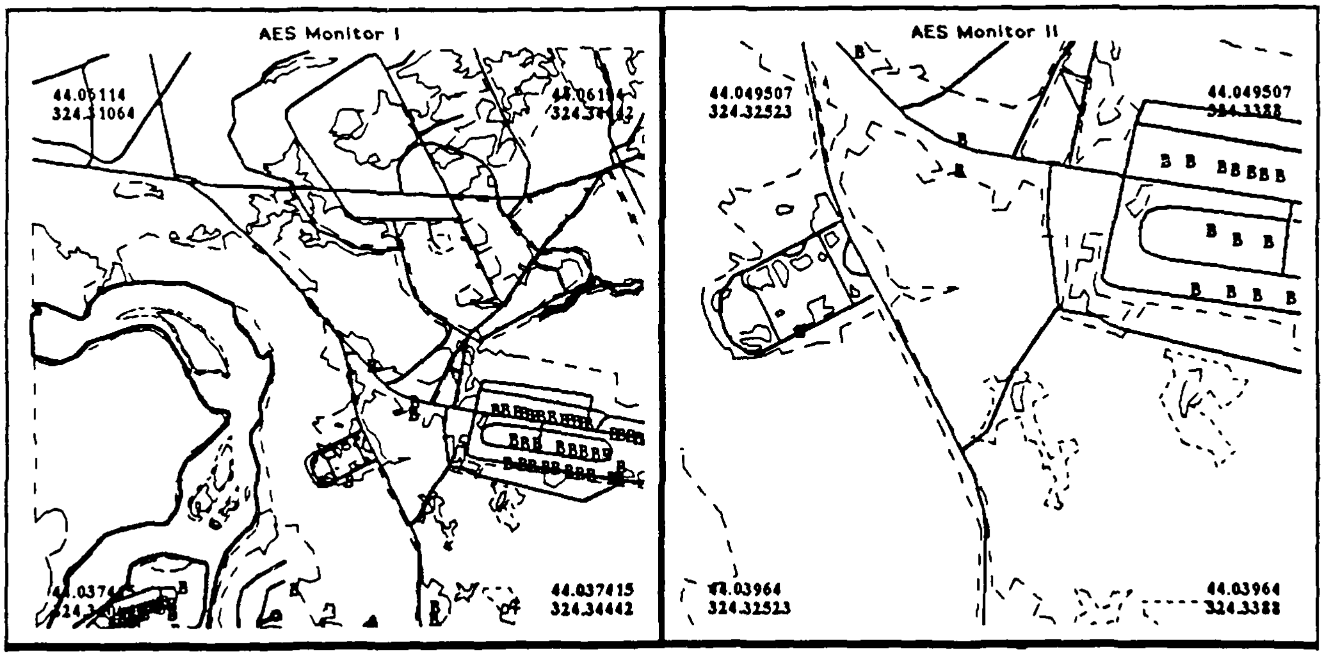


Figure 2. Terrain model for site-1: overview and close-up.

In addition to performing automated cue processing functions, the KBE provides interactive softcopy terrain modeling capabilities. Terrain data is introduced to the KBE via ASCII files containing terrain element descriptors. Capabilities within KBE's terrain functions include symbolic representation, display, and manipulation of Digital Feature Analysis Data (DFAD). The user can obtain or construct softcopy maps for areas of interest using terrain features appropriate for his application. Along with this, the KBE provides generic object-terrain reasoning capabilities for doing spatial reasoning. Areas of interest (where imagery was acquired) are modeled with digital terrain features such as roads, railroads, fresh water, mixed trees, building, and military construction.

The AES testbed includes capabilities for digitizing terrain features from imagery data in a semi-automated mode. Figure 2 shows a representative DFAD-like manually constructed terrain model for one area used in our examples. Currently, research is being conducted at Lockheed and elsewhere for automatically extracting terrain features and producing such symbolic maps from imagery data.

2.1.2 COBIUS

A constraint-based image understanding system (COBIUS) has been developed which focuses on high resolution aerial imagery interpretation. The major problems which this system addresses are:

1) generic domain object representation

2) compensating for unreliable image segmentation

3) knowledge control

The system consists of knowledge bases for domain object models and control strategies, blackboard areas to contain the instantiated hypotheses of objects and constraints, and an image feature database to fuse results from multiple image segmentation modules (Figure 3).

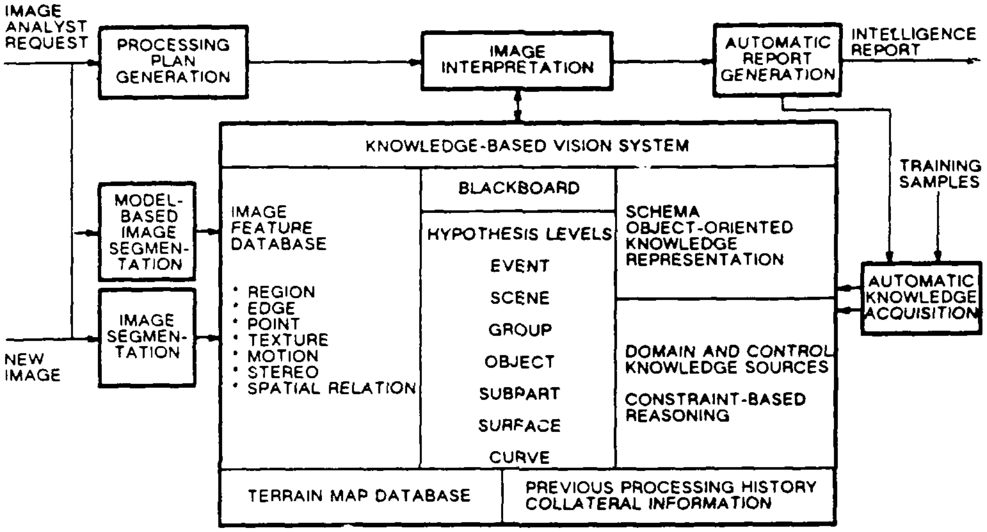


IMAGE ANALYST REQUEST
PROCESSING PLAN GENERATION
IMAGE INTERPRETATION
AUTOMATIC REPORT GENERATION
INTELLIGENCE REPORT
TRAINING SAMPLES
KNOWLEDGE-BASED VISION SYSTEM
MODEL-BASED IMAGE SEGMEN-TATION
IMAGE SEGMEN-TATION
NEW IMAGE
IMAGE FEATURE DATABASE
• REGION
• EDGE
• POINT
• TEXTURE
• MOTION
• STEREO
• SPATIAL RELATION
BLACKBOARD
HYPOTHESIS LEVELS
EVENT
SCENE
GROUP
OBJECT
SUBPART
SURFACE
CURVE
SCHEMA OBJECT-ORIENTED KNOWLEDGE REPRESENTATION
DOMAIN AND CONTROL KNOWLEDGE SOURCES
CONSTRAINT-BASED REASONING
AUTOMATIC KNOWLEDGE ACQUISITION
TERRAIN MAP DATABASE
PREVIOUS PROCESSING HISTORY COLLATERAL INFORMATION

Figure 3. COBIUS image understanding architecture.

To address problem (1) above, COBIUS uses a hierarchical representation scheme for both domain objects and constraints. The domain object representation hierarchy consists of event, scene, group, object subpart, surface, and curve levels. In a similar hierarchy, constraints are represented from coarse to fine at different levels. Both objects and constraints (at all hierarchical levels) are represented as schemas.

Constraints can be applied either to domain objects or other constraints. This allows the decomposition of complex constraints into primitive constraints and allows constraints to be modified by rules and other constraints. The advantage of this representation is that constraints can be treated like domain objects; therefore, model-based prediction and verification of primitive constraints from complex constraints can be used to reduce the combinatorial computation of graph matching techniques.

To address problem (2) COBIUS uses a multiple feature fusion approach with model-based feature verification capability. The image segmentation component consists of both region and edge segmentation modules, and a model-based resegmentation module that uses model information to guide the segmentation of expected objects in the selected image areas. The region segmention provides coarse image features that are useful for initial image interpretation. The edge segmention provides detailed shape information suitable for model-based verification.

Depending on the scene content, region or edge features are selected according to the current processing goal and strategy to provide adequate descriptions for feature-to-model matching. The model-based resegmentation routine is controlled by object verification rules. With the ability to fuse multiple image features and to use model information for resegmentation, COBIUS significantly ameliorates problems caused by unreliable segmentation.

To address problem (3) COBIUS reasons opportunistically, pursuing and refining the most plausible hypothesis first. Control knowledge is represented explicitly in terms of control schemes and strategy selection rules. Dempster-Shafer style uncertainty reasoning is used to minimize the problem associated with errors in the segmentation process. Uncertainty is also used as control information where high certainty hypotheses will be explored first. Another source of control information is encoded in the representation hierarchy of the constraints. This constraint hierarchy is used by the constraint manipulation rules to dynamically determine what are the best constraints to evaluate next.

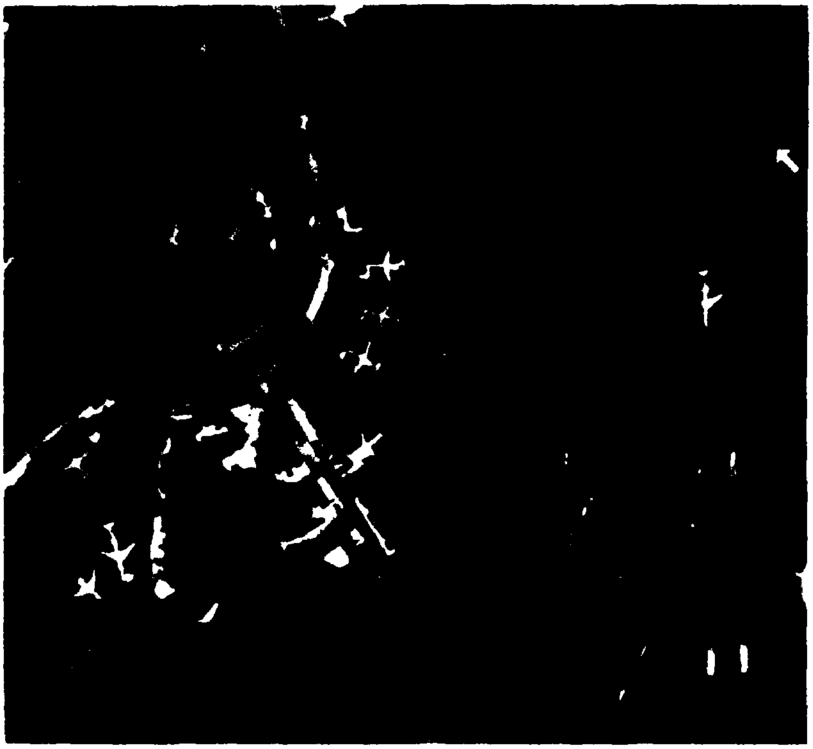


Figure 4. Split-and-Merge segmentation results.

COBIUS is implemented in Automated Reasoning Tool (ART) and LISP programming language on a Symbolics machine. The focus of our effort has been to design and implement COBIUS to be as scene independent as possible. As a processing example, however, we have applied it to airport scenes.

Region and edge segmentation results on an aerial photograph of Washington National Airport are shown in Figures 4 and 5. Long edge segments are grouped together to form runway hypotheses. Regions with straight boundaries and elongated shapes are classified as buildings. Constraints such as "buildings are not too close to runways" are checked to resolve inconsistent scene hypotheses. Building hypotheses satisfying the NEXT-TO spatial constraint form building-complex hypothesis at the group level on the blackboard. Additional building hypotheses are predicted at the two ends of the building complex based on the NEXT-TO constraint.

Once the building complex hypothesis has been extended, the airport scene model and constraints are used to predict roughly the locations of taxiways, parking lots, and access roads. These predictions help to narrow down potential areas of interest for detecting airplanes. Trapezoid shape primitives extracted from edge segments are used to form airplane subpart hypotheses. Spatial constraints between airplane subpart hypotheses are evaluated to form airplane hypotheses (Figure 6). Independently, regions with airplane model attributes are classified as airplanes. For partially supported airplane hypotheses, missing subparts are predicted (boxes in Fig. 6), and the model-based resegmentation module is used to verify those predictions. The verified airplanes are shown in Figure 7.

We have designed and implemented a constraint-based image understanding system for interpretation of aerial imagery and have applied it to an airport scene. The novelty of this approach is that we represent constraints as objects and organize them hierarchically (similar to the way objects in the scene are represented and organized). This allows tremendous flexibility for generation, combination, manipulation, evaluation, propagation, and satisfaction of constraints. In addition, it makes it convenient to adapt the system to new domains.

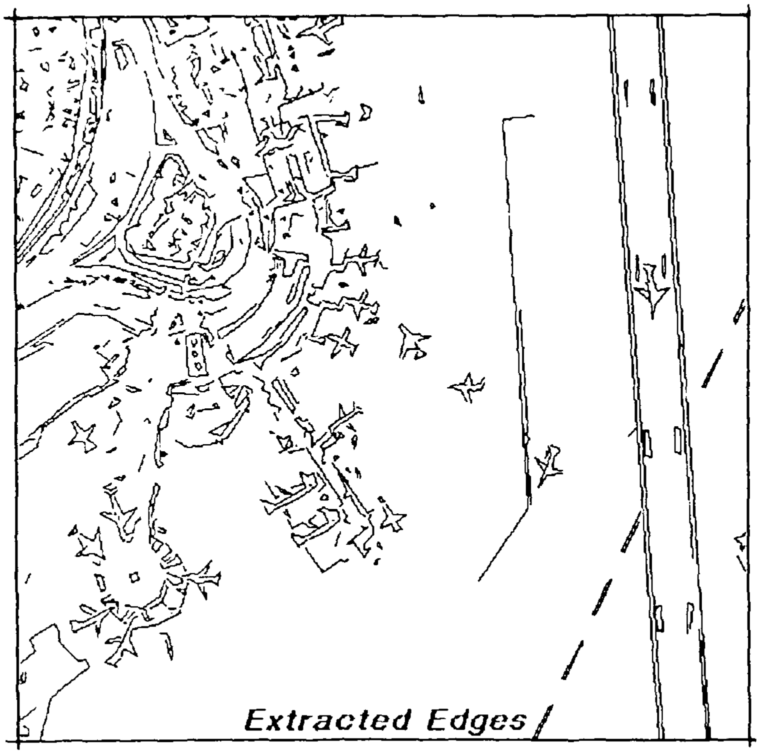


Figure 5. Canny edge segmentation results.

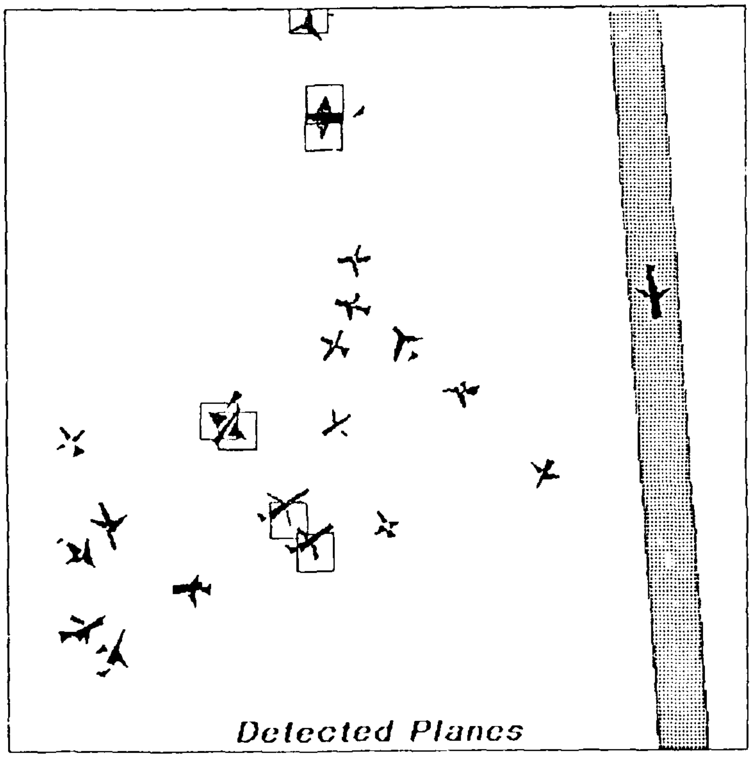


Figure 6. Detected airplanes and predicted wings.

We have also devised a set of flexible control strategies which can manipulate constraints in the same way as objects are manipulated. Using this representation scheme and control strategy, we provide greater modularity, flexiblity, and avoid graph matching and its combinatorial explosion as well.

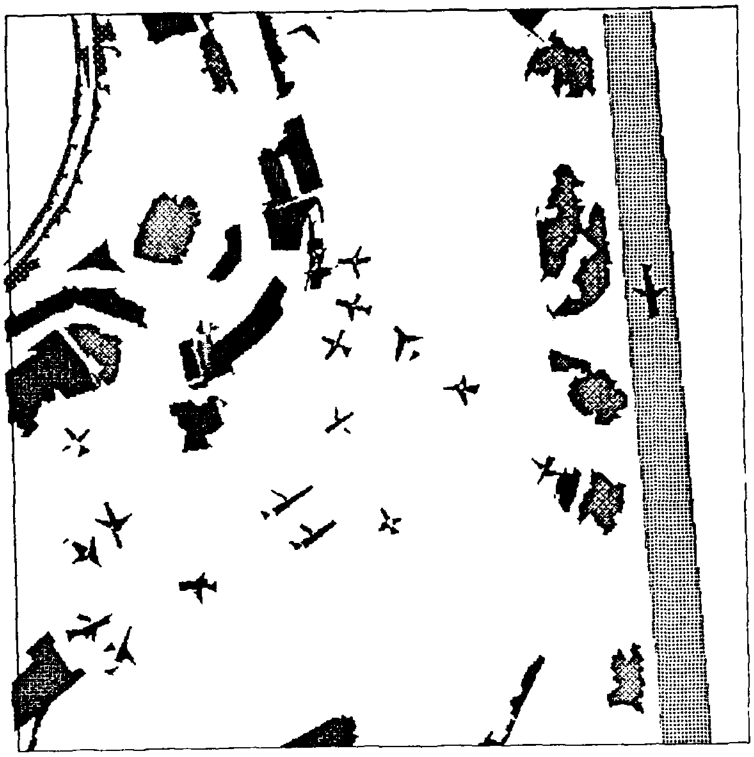


Figure 7. Final interpretation results.

We are currently exploring incorporation of the following schema into our image understanding system: (1) temporal relations and constraints to analyze sequences of aerial imagery, (2) ancillary knowledge from maps or previous scene evaluations, (3) detection of more´ sophisticated temporal events, (4) automatic report generation for monitoring results.

2.1.3 Cartographic image analysis

The overall objective of the Cartographic Image Analy: ،s project is to investigate image understanding algorithm methodologies which will provide accurate and efficient automatic interpretations of overhead digital (softcopy) imagery. This includes investigation of new automatic and semi-automatic techniques for feature (object) extraction and delineation, feature classification and identification, intelligent information storage and retrieval, image screening, and a variety of difficult application domain tasks such as change detection, trafficability analysis, and reference map preparation. Our effort explores new and promising technology areas which may yield fruitful results to the challenging aspects of automation for photointerpretation. Two recent highlights of this activity include: 1) a new tool called "High Accuracy Reference Map Extractor" which provides automated delineation capability of high-value urban objects with pinpoint accuracy, and 2) a new Neural Network for image understanding.

The High Accuracy Reference Map Extractor improves upon the results of the Lockheed semi-automated region extraction/delineation capability. It takes as input a noisy representation of an object's boundary and converts it to a sequence of straight line segments which yields an accurate object delineation. Such straight line segments are very appropriate for computer generation of reference data for a mission preparation task, with much improved results deriving from the application of this strategy. User efficiency and more effective personnel utilization can occur from the application of such a strategy.

The second highlight exploits multiple image resolution levels for improved delineations and classification of high-resolution digital imagery using a Neural-Network structure. A given feature is best represented as a whole at some optimal level of detail, dependent on its spatial dimensions (the recognition of a vehicle requires a higher level of detail than the recognition of the road it is on). At resolutions below the optimum a feature may not be visible, or contextual features may dominate. At resolutions above the optimum a feature's internal features may dominate. Information on either side of the optimal threshold is important for reinforcing or cancelling the interpretation of a feature, all of which is ignored in the more standard global interpretation process. A set of requirements for a new approach to image interpretation, called Multi-Pass Multi-Resolution (MPR) was developed which partitions refined shapes by splitting and remerging adjacent partitions according to the type and quality of their classification, and tracks the changing confidence in feature interpretations due to refinement of partitions and the exposure of internal features, weighted, or "fuzzy", classes. The preliminary results on urban imagery are extremely encouraging.

2.2 APPLICATIONS OF NEURAL NETWORKS

2.2.1 Automated terrain elevation extraction

We have studied how match points between stereo image pairs can be used to derive terrain elevation data from aerial photography. The applications are obstacle avoidance and target downtrack ranging for camera guided vehicles. These measurements are made possible by the relative image offsets, or parallax, produced when objects at different ranges are imaged from different angles. In other image comparison applications, such as change detection, parallax may constitute a significant nuisance, producing undesired relative image distortions. Parallax removal through pixel by pixel image matching is then necessary before image comparison can be performed.

Fractional parallax determination at each pixel location was attempted by using image cross correlation calculations. This data, corresponding to image window sections, was input to a back-propagation, two hidden-layer, symmetrical neural netwo . Network output included the parallax offset value for the center pixel of each window section. The network was trained using simulated stereo imagery so that the exact parallax offset at each pixel was known.

Network results on simulated test sets show a distinct improvement over results using correlation smoothing methods. The trained network was then used on a real image pair for elevation extraction and change detection. The resulting elevation surface and change detection difference image are illustrated and evaluated in section 3. Further results of this study are also described there.

2.2.2 Inference mechanisms in knowledge-based systems

In a second study, relations among neural networks, linear programming and continuous or "fuzzy" logical inference have been derived and demonstrated. A statement in the propositional calculus, like "$p \supset q$" may be represented as a function of its propositional variables, e.g. $f \supset (p,q)$. These functions are generally thought of as integer valued functions of the discrete variables p,q; for example, $p \supset q$ or $f \supset (p,q) = 0$ if $p=l$, $q=0$, or $f \supset (p,q) = 1$ otherwise. We have explored the consequences of generalizing these functions, allowing the propositional variables to assume a continuum of truth-values $0 \le p \le l$, $0 \le q \le 1$, in such a way as to coincide with the traditional functions of discrete logic at the corner-points of the (p,q) square. We call the resulting theory "continuous logic".

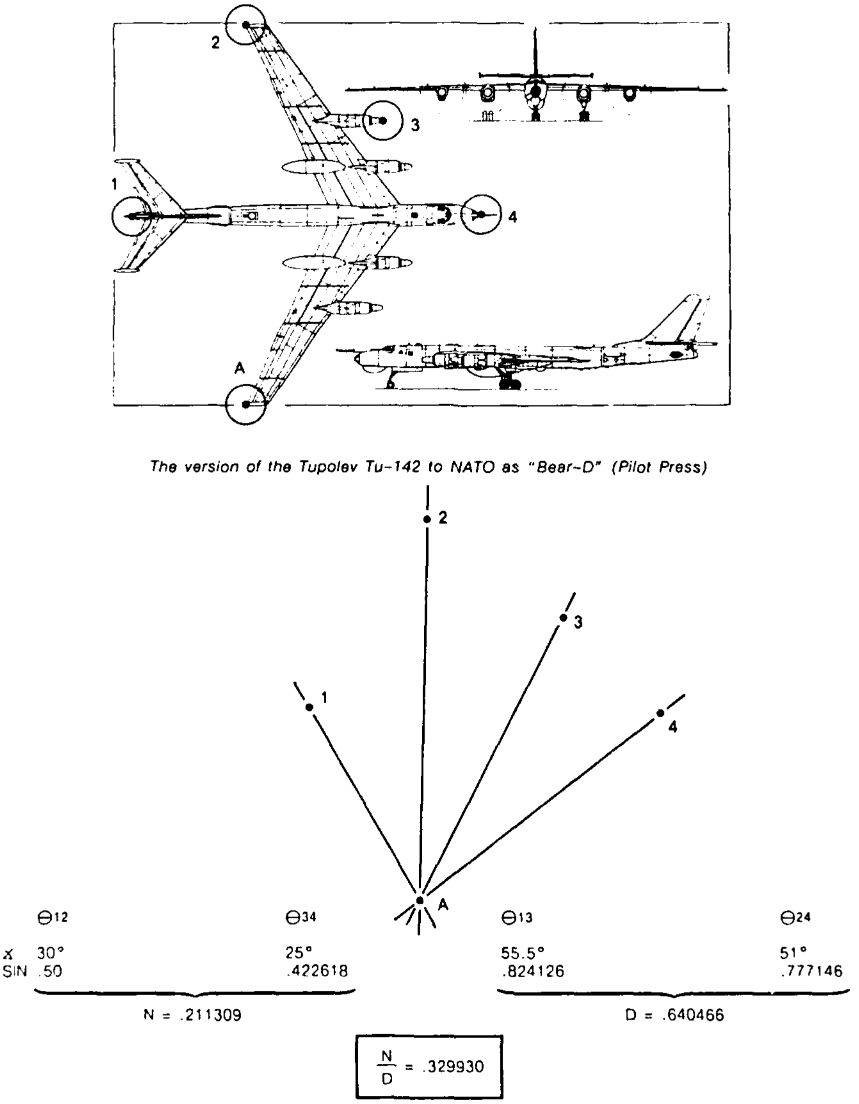
Theorems about the logical functions $f \supset (p,q)$, $f_v (p,q)$ $f_{.}(p,q)$ $f_{\bullet}(p,q)$ can now be provided by reasoning about mathematical inequalities. A tautology in this theory is a statement whose minimum truth value, subject to the mathematical inequality constraints imposed by the premises, is unity. The proof of a proposition becomes a mathematical exercise in non-linear or linear programming, maximizing functions of propositional arguments subject to inequality constraints. If the logical functions are defined as linear algebraic functions, proving theorems in logic is accomplished by solving linear programming problems, which becomes a candidate inference mechanism for knowledge based systems grounded in continuous logic. The linear programming procedure generates a monotonic path through the search space; backtracking or trial-and-error are not required. With this theory, we can explore the consequences of attributing truth-values other than 0 or 1 to propositions, and we can prove statements involving generalized logical relationships among such propositions.

The second element of our approach makes use of J. J. Hopfield's observation that a variety of constrained optimization problems, including linear programming, can be solved by neural networks. We have shown how to concatenate these results, and have demonstrated several examples of solving logical inference problems with neural networks. Applications of these results to automated inferencing in "smart" collateral data bases are being investigated.

2.3 PERSPECTIVE AND MOTION INVARIANT OBJECT RECOGNITION

2.3.1 Projective invariants

Dimensionless metrics derived from ratios of geometrical distances and volumes, may provide a computationally low-cost means for classifying and cataloging objects in imagery. The cross-ratio theorem in the plane is an example of using the ratio of sines resulting from object feature locations in image space to perform image rectification and object identification. After establishing the basic invariant theorem under Mobius transformations, the principle of image space property invariance is generalized to arbitrary scene space point placement, then extended towards stereoscopic and unconstrained object classification in higher dimensional space. Particular focus is paid to the 2-1, 2-2, and 3-2 scene-to-image dimensional cases and the forms which these invariants assume. Several examples of applications of invariants in object identification have been demonstrated, using engineering drawings and "ground photography" of aircraft and vessels. Figures 8 and 9 show an example of comparing invariants calculated from engineering drawings of the Bear-D bomber, with invariants measured from an oblique photograph of this aircraft.



The version of the Tupolev Tu-142 to NATO as "Bear-D" (Pilot Press)

Figure 8. Bear-D engineering diagram with calculated invariance.

2.3.2 Motion invariant image features

A method has been developed which uses correspondences of five lines in two views of a moving object to calculate its structure, axis of rotation, amount of rotation, and translation vector. If the observer is moving, then the structure of the environment and the motion of the observer is determined. The only information used is the directions of the matched line segments between two views, i.e., no point to point correspondence is required. This alleviates the feature matching problem, since it is generally easier to match line segments accurately between two pictures than to match points.

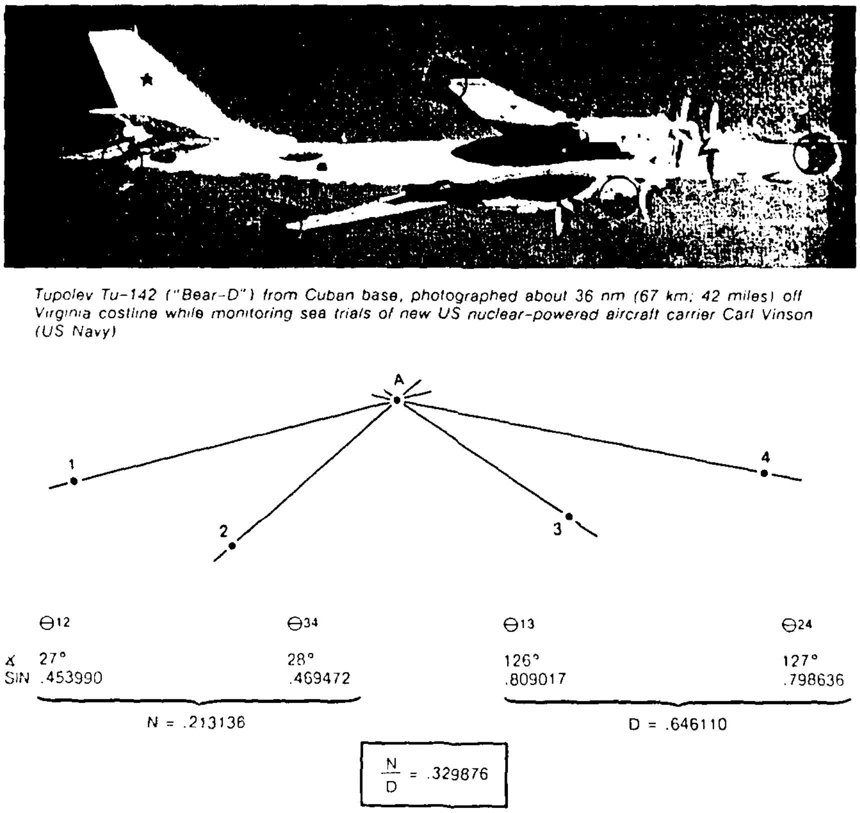

Tupolev Tu-142 ("Bear-D") from Cuban base, photographed about 36 nm (67 km; 42 miles) off Virginia costline while monitoring sea trials of new US nuclear-powered aircraft carrier Carl Vinson (US Navy)

Figure 9. Oblique photo of Bear-D with calculated invariance.

Most published techniques for motion estimation and determination of structure from motion assume the moving object (and/or the environment) is rigid. This assumption is often implied and sometimes is used explicitly to establish equations. A few researchers have used line correspondences along with the explicit use of the rigidity assumption.

In the Lockheed method, the structure of the moving object (i.e., the relative position of the line segments in space) is determined first by solving a set of polynomial equations. These equations are based on the following property of a rigid object: the angle and distance between any two lines on the object do not change from frame to frame. Once the relative positions of the lines in space are recovered, the axis of rotation, the amount of rotation, and the translation vector are easily recovered (in that order) by solving a set of linear equations for each parameter.

2.4 SUMMARY OF OTHER IMAGING RESEARCH

2.4.1 Space astronomy

Research has been carried out in the development of x-ray, extreme ultraviolet, and ultraviolet sensors to gather diagnostic information about energetic physical processes occurring amongst space bodies. In this environment it is important to understand the natural space photon environment and its role as a source of background noise. Edge enhancement, threshold, noise reduction, and segmentation algorithms have been developed to interpret the spectral signatures returned by these extreme wavelength sensors. These algorithms have contributed, for example, to an understanding of the behavior of supernova 1987a.

2.4.2 Material science

Advanced polymer-based materials used in aerospace systems are typically very complex, possessing a wide variety of poorly understood chemical and physical interactions, both at the molecular and macroscopic levels Research has centered on nuclear magnetic resonance (NMR) spectroscopy and NMR imaging to investigate these interactions. NMR imaging has provided a tool whereby critical variations in important features such as local particle variation of solid propellants can be mapped out. Simulants containing a broad range of materials have been studied for the detection of embedded voids and particle agglomerates. The technique employed has been projective reconstruction using only one spatial encoding gradient. This approach, first used in the medical sciences, reduces the imaging time in half, but with the slight disadvantage of a greater propensity of ghost images. Defects are located in the NMR images by simple thresholding and edge enhancement processes. Projective reconstruction has also been employed to determine the behavior of viscous materials undergoing laminar flow. Contrast differencing techniques have been used to chart the unidirectional movement of suspended particles in NMR temporal image sets.

2.4.3 Integrated electronics and architectures

We have investigated how image analysis algorithms can be implemented on neural networks, and the related development of a massively parallel hardware system called GSIMD using very large scale integrated (VLSI) chips. The architecture which has emerged contains a smart controller and a large configuration of Geometric Array Pixel Processing (GAPP) single-bit array processing chips assembled onto standard size 19-inch rack-mount boards. The system has facilitated the development of near real-time algorithms for aerial surveillance, visual inspection, and robotic vision.

2.4.4 Advanced software

An important question to consider in the implementation of IU algorithms is how shared memory Multiple Instruction Multiple Data (MIMD) parallel architectures can be applied to speed-up results. Our research has concentrated on practical implementation of algorithms for region-based segmentation, stereo topographic profiling, and image compression. These algorithms are designed to run on commercially available multiprocessors, e.g., Sequent B/21 and Alliant FX/80. Approaches considered for segmentation include tile-seaming, straight region growing amongst mutually cooperating processes, simulated annealing, and successive relaxation. Application of a specific technique depends upon the size of images, qual of segmentation desired, and absolute (real-time) speed requirement..

Table 1. GSIMD array processor benchmarks.

Operation	Time GSIMD	Time Other Systems
2D Convolution	15 µs	30 ms (Cray)
11x11 Convolution	0.2 ms	3 ms (CM2)
Sobel Edge	30 µs	30 ms (Cray)
Gaussian Zero Crossings	0.25 ms	0.5 ms (CM2)
Isolated cluster elimination	7 µs	5 ms (Cray)
Neural Net 100x100 Forward & Back	6 ms	1 s (Cray)
Random Assignment Problem	0.5 ms	1.5 s (Cray)
Abingdon cross benchmark	0.1 ms	1.5 ms (CM2)
Voronoi Diagram Construction	0.15 s	0.5 s (CM2)

Numerous SIMD algorithms have been devised and coded for the GSIMD array processor. Table 1 contains some benchmarks for these algorithms and comparisons with the Cray-2 and Connection Machine (CM2).

2.4.5 Solar physics

Solar phenomena modify and control the Earth's atmospheric and magnetospheric environments. Predicting and understanding solar flares, magnetic storms, and solar particle events are essential to the survivability of man and satellites in space. An ongoing project is attempting to investigate these phenomena by exploring the use of optical disks for the storage of large temporal image databases. With this technology, exceptional continuum frames from the Swedish Solar Observatory have been processed to make the highest resolution solar movie in existence. Transverse flows have been measured in this data by correlation tracking, and a vortex flow in the photosphere has been discovered. This is the first reported observation of such vortex motion in photospheric levels, although numerical simulations of granular convection have predicted this behavior. Fourier filtering of this movie has removed the variability caused by observatory seeing to a remarkable extent.

2.4.6 Nondestructive-Testing Technology

Improvements in hardware systems are often achieved by reducing tolerances in design. Nondestructive testing (NDT) is a general class of inspection methods that leaves objects unharmed and suitable for their intended use, yet ensures that hardware meets designed tolerance levels. Research carried out by the NDT group has concentrated on both noncontact ultrasonic and x-ray scatter imaging techniques. Significant progress has been made in (1) correcting realtime radiographs (RTR) for gain and offset using digital filtering techniques, (2) identifying material composition using dual-energy RTR imaging, (3) demonstrating the feasibility of noncontact ultrasonic inspection, and (4) developing the first digital laminographic technique that uses RTR images to focus on selected depths of an object undergoing radiographic inspection.

2.4.7 Automation and robotics

Over the past several years a sophisticated testbed has been developed to support applications such as the Mars rover, Space Station, and satellite repair and recovery. This dual arm system consists of a network of Intel 80386 microprocessors with built-in control algorithms, advanced Datacube machine-vision hardware and software, conventional arm sensors for measuring displacement, velocity, acceleration, force, and electronic currents, special ERIM range-finding sensors, and communication links for remote command. Vision cameras are mounted on both wrists for close-up work, and twin stereo cameras are mounted on the robot body for navigation and collision avoidance. To simulate space-based activity a special laboratory has been furnished with an air table, special lighting fixtures, a safety system, and a operator command center. IU research has focused on object modeling and recognition, obstacle detection, and collision avoidance algorithms.

3. DERIVATION AND VISUALIZATION OF 3-D DATA

The preceding overview was intended to show the very wide range of image exploitation activities underway at Lockheed. The rest of this paper highlights a specific technology arena, that of 3-D scene exploitation. Lockheed has been exploiting the characteristics of 3-D scene knowledge for enhanced image understanding capabilities for more than 10 years. Accurate 3-D scene knowledge can provide significant advantages for more reliable image modeling and interpretation applications. Recent advances in parallel processing has also provided enhanced capabilities in several areas where speed has been a concern. We describe examples below which illustrate activities in 3-D laser ranging, 3-D topographic reconstruction, and 3-D scene visualization.

3.1 LASER RANGING APPLICATIONS

A laser range imagery sensor provides accurate distance measurements to a raster of scene locations. Such data can provide significant insight for a variety of image exploitation problems. An obvious application is for terrain and obstacle avoidance. Resolution and accuracy of operational range sensors now available combined with advanced 3-D algorithms have also yielded the development of highly reliable tactical mobile targeting, guidance position updating, and offset aimpointing capabilities. Two application areas are described.

3.1.1 Laser Ranging Data for Offset Aimpointing

A highly accurate surface shell representation of the viewed scene is extracted. We have exploited such imagery to identify and extract planar surfaces associated with building walls. The relative locations and orientations of these surfaces are then used for precise offset updating.

The reference information required is the building orientation and its computed normal distance to the offset location. If the offset location lies at the intersection of two building walls, each of their offsets is zero. More generally, the offset is the shortest (i.e., perpendicular) metric distance from the offset point to the infinite plane through the building wall. Planar surfaces and their normal distances to the sensor are easily extracted from the sensed data using Hough transform techniques which determine the Cartesian (x, y, z) values associated with the pixels on the building walls to identify both the 3-D planarity and constant normal distance values as constraints.

The vehicle location is computed as a simple mathematical functions of the metric difference between the reference and sensed normal distance pairs. A minimum of three non-coplanar surfaces are required for a precise 3-D geodetic result. Figure 10 shows typical normal distance values extracted. A least-squares solution is computed when multiple walls are available. The derived metric value of the normal distances between sensed co-planar wall surfaces are used in conjunction with their stored values in the reference data base to match sensed building wall surfaces to their corresponding reference wall. Tests on both sensed and synthetically created data show very precise aimpointing accuracies.

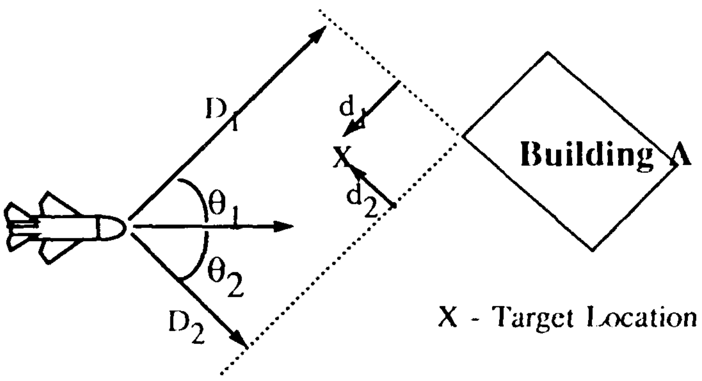


Figure 10. Example of normal distance values between building and offset locations.

Reference preparation requirements are greatly reduced over many traditional position updating strategies (e.g., correlation) since: 1) only a limited set of surface features and their attributes are required and 2) accurate scene rendition is unnecessary. Preparation for the mission of interest can proceed from maps which show the relative positions of buildings in the scene used for aimpoint or guidance updating. Geodetic errors would then only occur due to: 1) the extent that the map has mislocated the buildings on it, and 2) the later accumulation of vehicle position, orientation, and velocity errors. Figure 11a shows a synthetically created image of a building scene; Figure 11b shows the results of matching the sensed graph associated with the scene wall extractions to the reference graph which would be stored preflight

3.1.2 3-D Object Modeling

Another area of interest is the recognition of 3-D objects for robotic applications. Such recognition can be considered a two-stage problem. In the first stage, the system analyzes the object model to determine prominent object features suitable for recognition and to compile these features into an algorithm for object recognition. In the second stage, the system applies the selected algorithm and recognition strategy at run time for efficient object recognition. Most existing IU systems only address the second stage problem. The tasks of identifying prominent object features and recognition strategies are left to the system developer who manually encodes this information into the system.

We plan to automate the first stage problem by developing an object model feature compiler that extracts and organizes object features automatically from object models based on geometric properties and spatial reasoning. In our representation, object recognition is treated as a general graph matching problem where image features are matched with object model features. The function of the object model feature compiler is to identify prominent features from object models and order them according to their importance to form an efficient recognition strategy. The object model feature compiler accepts geometric models created for CAD/CAM purposes and also takes advantage of domain specific rules to determine recognition strategies..

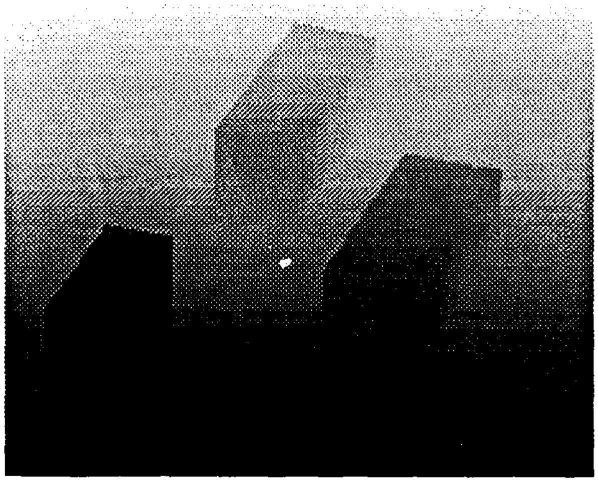


Figure 11a. Synthetic Building Scene.

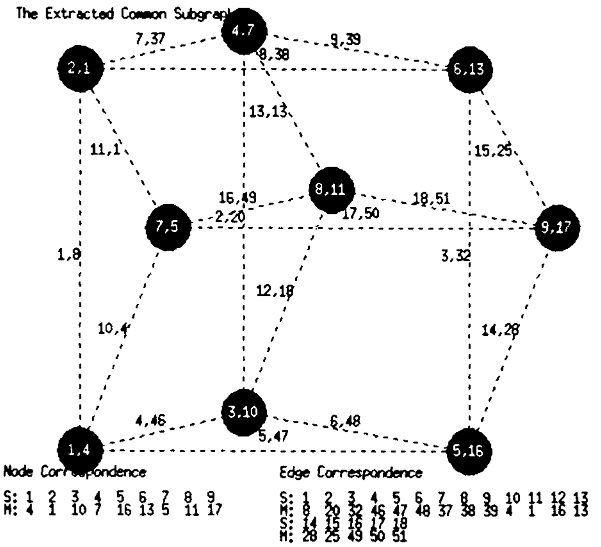


Figure 11b: Results of Graph Matching

The object model feature compiler generates two types of object feature graphs--qualitative and quantitative. The qualitative object feature graph describes the general structural configuration of an object. This graph is used to recognize a generic object class and to determine coarse object recognition. For each qualitative graph, a more detailed quantitative graph is used to recognize the specific object instance and to find the exact object orientation.

The object model feature compiler does not need to generate all the features, but just enough to achieve reliable object recognition. Therefore, the size of a compiled algorithm is much smaller than information needed to display the object model. The purpose of the object model feature compiler is to select the minimum number of nodes and links in the graph that provide sufficient information to recognize the object.

3.2 3-D TOPOGRAPHIC RECONSTRUCTION

3-D sensed data is available in a very limited set of domains. More conventionally, stereo pairs are used to determine topographic properties of the sensed imagery. Match points between the stereo pairs are determined, and the relative image offsets, or parallax, used to determine terrain elevations. We describe two algorithm strategies, one area-based and one edge-based, which provide accurate 3-D reconstruction of such data.

3.2.1 DTED Perspective View Generation

In Lockheed's area-based approach, correlation proceeds from coarse to fine image resolutions. Parallax data from coarse resolution data is used to dewarp the right image to the left image at image resolution 2X finer than the coarse imagery. Parallax offsets at the finer resolution are bilinearly interpolated from the first stage grid values. This operation is typically repeated until the finest resolution is reached.

Coarse image registration can be achieved by matching image windows from one image to another using a correlation search. A window of pixel grey levels from one image is stepped along a range of offset positions along the parallax direction in the second image. The cross-correlation of pixel grey levels is calculated at each position. The peak correlation position determines the average parallax offset over the image window.

The initial pass (pass1) involves repeated calculation of the two dimensional cross correlation formula:

$$\rho = \frac{n\Sigma x_i y_i \ - \ (\Sigma x_i)(\Sigma y_i)}{[n\Sigma x_i^2 \ - \ (\Sigma x_i)^2][n\Sigma y_i^2 \ - \ (\Sigma y_i)^2]} \qquad (1)$$

where the x_i and y_i are window intensity samples from the left and right views and n is the number of these samples. Two samples of size n are said to be well correlated if the correlation coefficient ρ approaches the value 1, where $-1<=\rho<=1$. Implementation of this step can be carried out by applying a sliding window concept to compute the Σy_i and Σy_i^2 in the right view. Depending upon the size of the overlay window, a significant savings in the total computation time involving formula (1) can be made by precomputing the integrated right view, and its pixel by pixel square. We have investigated the use of Vector and Multiple Instruction Multiple Data (MIMD) machines for this step and have shown that significant speedup ratios can be realized. Table 1 gives some of the performance speedups obtained. Figure 12 contains a block diagram of the basic algorithm. The term FOM should be interpreted as "Figure of Merit."

Table 1. Pass1 benchmark on Sequent B/21.

number of processors	time (min.)	speedup ratio
1	69.98	-
2	35.90	1.95
3	23.93	2.92
4	17.95	3.90
5	14.95	4.68
6	11.97	5.85
7	10.47	6.68
8	8.97	7.80

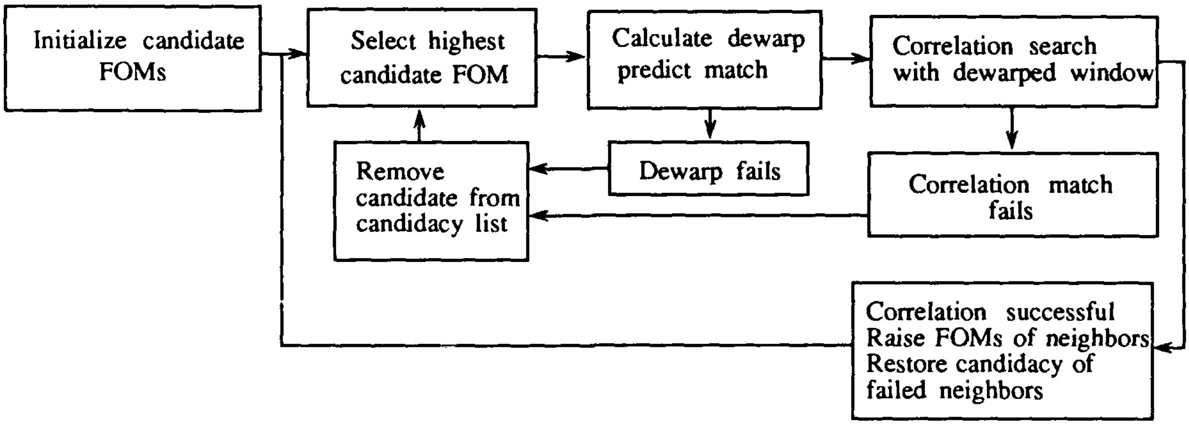

Initialize candidate FOMs
Select highest candidate FOM
Calculate dewarp predict match
Correlation search with dewarped window
Remove candidate from candidacy list
Dewarp fails
Correlation match fails
Correlation successful Raise FOMs of neighbors Restore candidacy of failed neighbors

Figure 12. Pass2 dewarp correlation algorithm.

A neural network has also been implemented to estimate the match position to subpixel accuracy based on the image correlation trends in the vicinity of the pixel location being matched. It provides increased capability where sharp variations in parallax offset (terrain relief) exist, which can cause deterioration in correlation value related to the match position variance. The statistical relationship of correlation as a function of image position (x, y) and parallax offset (z) can become rather complex. Neural networks are well suited to learning these statistical relationships using training data for which the correct parallax is exactly known.

The input to the neural network consisted of a volume box of correlation data containing the correlation values for an image window centered at a (x,y) pixel position in the first image and correlation values for the $-5 < x < 5$ neighboring pixels. The correlation values at each position span a search range of parallax offsets (locally proportional to terrain elevation) forming the z-dimension of the box. Each step in z is one pixel of parallax offset.

The output layer of the network is a single neuron which calculates, as a function of its inputs, a value representing the parallax offset at pixel (x_0,y_0). Between the input and output layers are "hidden" layers. Each input signal value is multiplied by a weight value and then the weighted inputs are summed. A bias value for the neuron is added to the sum. The performance of the net is determined by the values of the weights and biases which are adjusted in the training process. The training data set was obtained by correlating images of a simulated stereo pair. The grey levels of the first image were generated using a damped random walk. Elevation data were generated by the same method and then the original grey levels were modulated to simulate sun shading effects according to terrain slope. The second image was generated from the first using a "painting" method to accurately simulate parallax distortion according to the elevation variation.

Table 2 summarizes the comparison of neural network performance versus the traditional adaptive correlation smoothing method. The data represented quite rough terrain. In addition, small amounts of noise were introduced in one of the images. The data used spanned a test set of 18,000 parallax match points which did not include the training set.

Table 2. Neural network performance on simulated data.

Method	RMS error in pixels	99th percentile error	% matches with error > 1.4 pixels
unsmoothed	1.14	4.6	11.8
adaptive	.75	2.9	5.9
neural net	.55	1.8	2.7

The neural network results show a significantly reduced RMS error in parallax determination. In addition, the table shows that the network was able to significantly reduce the number of larger errors.

The network, trained on simulated imagery, was applied without further changes on a selected real image pair. The images chosen were vertical photographs taken from an airplane near San Bruno, California. The difference in viewing angle is 30 degrees. The stereo pair, reduced in size for viewing by stereoscope, is shown in Figure 13.

The images were then processed by a Lockheed developed automatic terrain elevation extraction program to obtain about a hundred coarsely spaced high confidence tie points. These tie points were used to accurately determine parallax direction and cross-parallax relative image distortion. The images were then warped to produce an image pair with the x axis along the parallax direction and excellent image registration in the cross-parallax y direction (a full quadratic dewarp was adequate for this image pair).

3.2.2 Urban Scene Stereo Reconstruction

Urban scenes provide a different type of environment for stereo disparity computations as the terrain does not vary smoothly and continuously. Edge-based matching techniques, unlike area-based approaches, match edge points which correspond to surface discontinuities. This means that the edge location in an image can be detected to sub-pixel accuracy. Consequently, the accuracy of height estimation is higher than that obtained from area-based matching. Another difference is computational efficiency. The search space for edge-based matching, consisting only of edge points, is a considerably smaller subspace of the entire image. Thus, the computation time for edge-based matching is less. In fact, the search space can be further reduced if one knows the geometric relationship between the cameras used in taking the stereo pair. That is, once the pair of stereo images are rectified so that the epipolar lines are horizontal scanlines, a pair of corresponding edge points in the left and right images need only be searched within the same horizontal scanlines. If the camera parameters are not known, the geometric relationship can be estimated.. When the two principal imaging planes are vertical to the ground, the estimation is performed by using a four-parameter transformation which accounts for rotation, translation, and scale change. With a set of tie points selected from the stereo pair, the parameters of the transformation are estimated and then used to warp the stereo images individually so that the epipolar lines are horizontal scanlines. The pixel intensity values in the geometrically transformed images are determined by cubic interpolation. Once the images are warped, the next step is to extract edges for stereo matching. Any of the currently available edge detectors such as the Laplacian of Gaussian zero-crossing technique or the Canny's operator can be used to extract edges. We used a multi-resolution segmentation technique to segment the image into regions because: 1) the edges can readily be extracted from regions, and 2) the region information can be used in conjunction with height information for other investigations such as feature classification and interpretation.

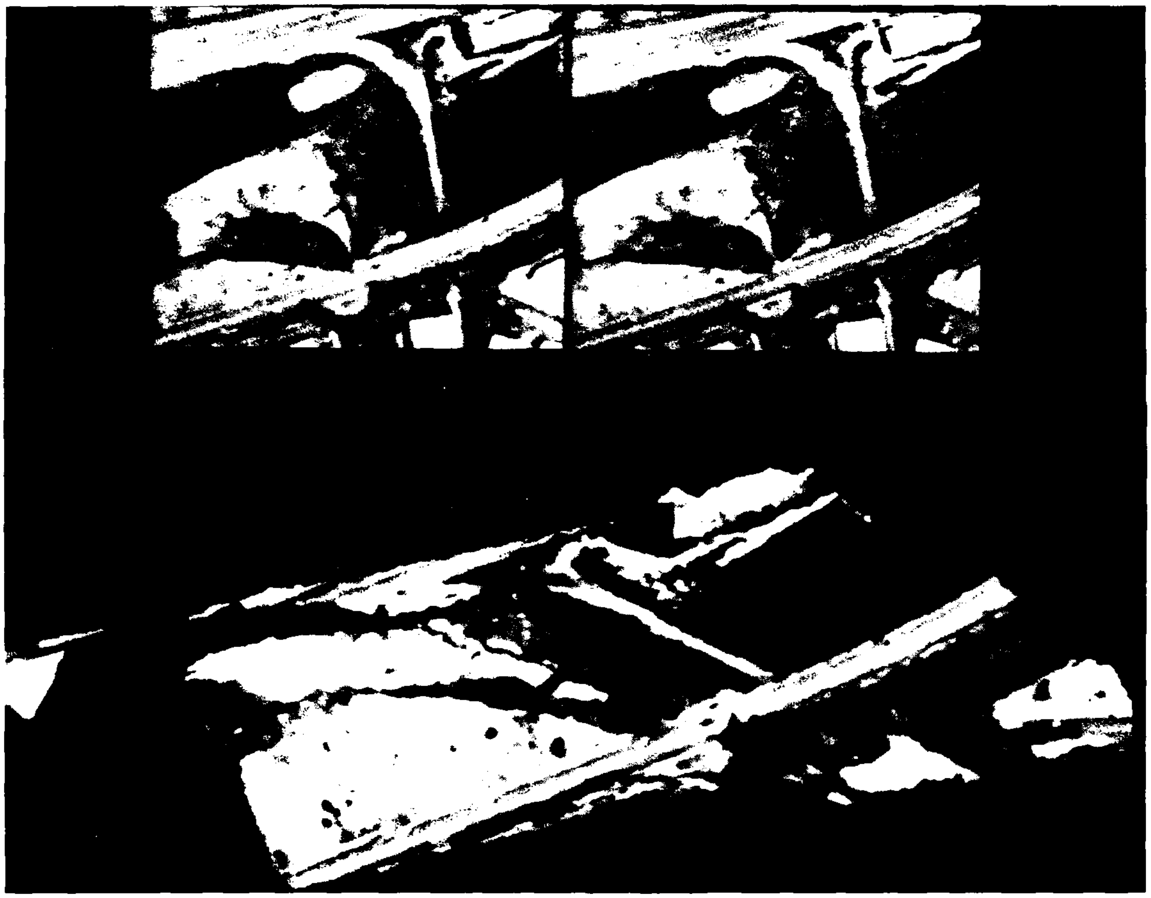


Figure 13. Top: Stereogram of test scene.
Bottom: Perspective reprojection of scene using neural network derived elevations.

The process of matching edge points is based on intra and inter-scanline search. Intra-scanline search refers to the problem of finding the pair of corresponding edge points within the same horizontal scanline in the left and right images. Inter-scanline search refers to the use of mutual dependency between scanlines as consistency constraints. In intra-scanline search, the edge-delimited intervals are used as the basic elements for matching. Using the nonreversal constraint in edge correspondence, namely, the order of matched edges has to be preserved in the left and right scanlines, the search is done in the left-to-right order on each scanline. The intervals are matched based on the similarity of the mean and the standard deviation of the intensities for the left and right intervals as well as the starting and ending pixel locations of the intervals. When the best match is found, disparities of the starting and ending pixels for the matched pair of intervals are calculated. When no good match can be found for an interval, the interval is ei'her skipped or merged with its neighboring interval. Skipping intervals corresponds to the case where a visible part in one image is occluded in the other image. Merging intervals corresponds to the case where some edge pixels may in fact be noise pixels and therefore have no matching points on the conjugate scanline.

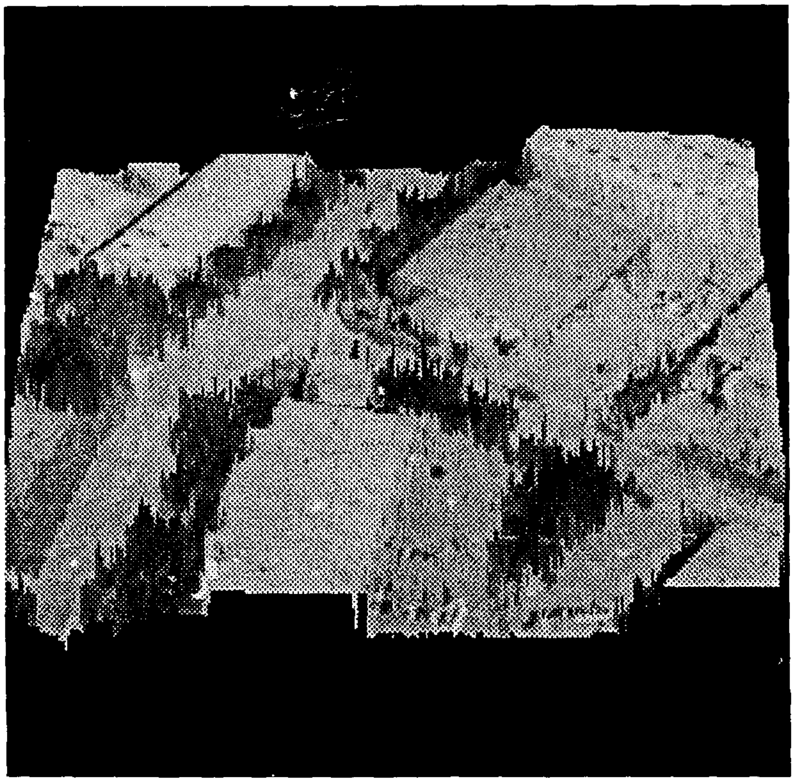


Figure 14. Building shapes from stereo images.

After the intra-scanline search is performed for each pair of scanlines independently, the inter-scanline search is processed across the scanlines to detect and correct the matching results which violate the consistency constraints. The intra-scanline search performs a local optimization for the correlation of individual lines in the image. A strong global constraint is apparent from the fact that the connected edges of a region delineated by the multi-resolution approach should have similar disparity values. Therefore, a pairing of edges is inconsistent if its disparity is greatly deviated from the average disparity of all pairings in that region. Using this connectivity assumption, inconsistent edge pairings are removed. A rematch is made for those pixels using the average region disparity to guide the search. In a similar fashion, gaps along the edges are filled. To produce a complete disparity map, the following simple interpolation scheme is used. For each pixel in the interior of a region, two linear interpolations are done. One interpolation is done between the two closest edge pixels on the opposite sides on the horizontal line. The other interpolation is done vertically. The average of the two interpolated values is assigned as the disparity for that pixel. Inter-scanline search is again applied to remove and correct the interpolation results which violate the continuity assumption. Figure 14 shows the results of this processing on an urban scene pair.

3.3 3-D SCENE VISUALIZATION

3.3.1 Digital Perspective Generation and Display of Terrain Data

A combination of DTED, ocean sounding data, and aerial photographs were combined to form stereoscopic perspective imagery. Impiemented on a fast PIPE architecture permitted near real-time fly-through movies of a scene.

The process of rendering perspective views is well-known. Researchers at Lockheed have extended this capability by incorporating sonar data into a single view of both on and off-shore elevations. Ocean depth sounding data, such as obtained from NOAA (National Oceanic and Atmospheric Administration), consists of a list of features that describe individual points that make up a depth map. Each line in the list contains the latitude, longitude and depth, as well as a variety of other descriptors that pertain to a single sounding data point. The data spacing varies irregularly over the depth map, with points more densely clustered in regions of higher depth variation. This data must be manipulated to create a regularly spaced grid to correspond to the DTED configuration. A "gridding" algorithm was created which interpolates the value at each grid point by weighting surrounding depth values according to their proximity. The only depths included in each interpolation are those which fall within a rectangular window which is centered on the grid point. The size of this window is adapted based on data density. When no data falls within the maximum window size, the grid value is set to zero depth, accounting for land regions. The resulting grid is a smoothly varying, regularly spaced depth map with maintains the accuracy of the original sparse NOAA map.

A USGS map and a USGS aerial photograph of the composite area were digitized, and the photograph and sounding grid registered to the map using a two dimensional linear warp, maintaining a map coordinate reference. The land regions of the USGS data were then overlaid with the ocean bottom regions of the NOAA data, thus creating a composite image and a composite elevation grid. Perspective views of the elevation data could then be created.

The original perspective program, written in FORTRAN on a Sun 3/160, takes four minutes to generate a single 512x512 frame. A version of this program was adapted to run on an Aspex PIPE computer. A modified perspective reprojection equation was used:

$$\left|\begin{array}{c} x' \\ y' \\ z' \end{array}\right| \quad \left|\begin{array}{ccc} \gamma & 0 & 0 \\ 0 & \gamma & 0 \\ 0 & 0 & 1/r \end{array}\right| \; M \; \left|\begin{array}{c} x-x_0 \\ y-y_0 \\ z-z_0 \end{array}\right|$$

(x_0, y_0, z_0) is an aim point in space along the optical axis;
r is the range from the camera to the aimpoint;
γ is the overall magnification.

The parameters γ and r, along with the tilt and azimuth angles which are contained in M, govern the orientation of the reprojected plane. These four parameters are individually controlled through a programmable constant on the PIPE, enabling dynamic control of the reprojection. This new tool allows interactive generation of fly-through movies of a scene.

This display capability provides a strikingly effective way of enabling an analyst to view the data in a context that is familiar and easily recognizable. The viewer can also readily relate coastal surface terrain features to their sub-surface counterparts. Figure 15 is a stereo pair of coastal terrain images prepared on the PIPE computer using techniques described above.

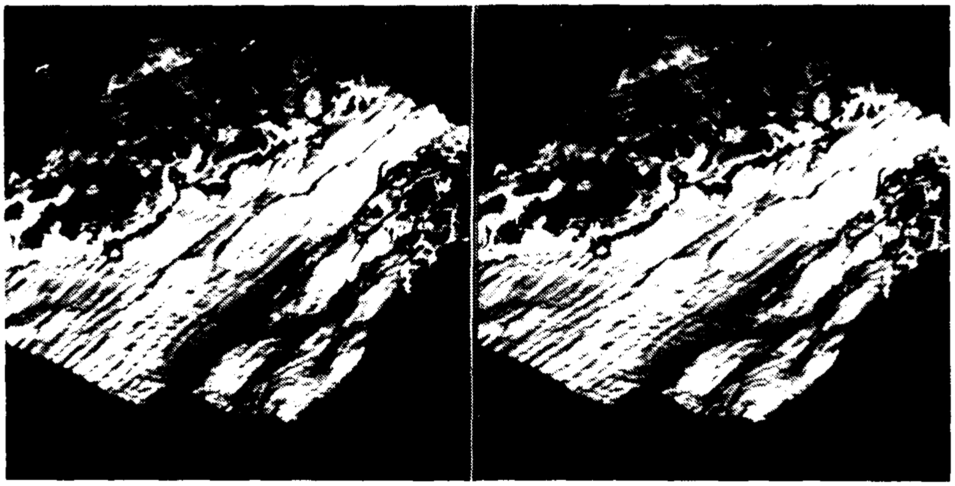


Figure 15. Land and ocean bottom coastal terrain.

3.3.2 3-D Visualization Research

Volumetric 3D data sets arise from several sources, such as stacks of adjacent computed tomograms and fluid flows computed or measured on a 3D sample grid. Such images can provide significant information about structures embedded in the dense array of samples. Because of the overwhelming quantity of data, understanding such images requires the extraction and interactive display of important structures. We have investigated this problem for scalar fluid flow data.

The first step is to extract interpolated iso-intensity surfaces represented by voxels, (small rectangular parallelepipeds). A contour surface can frequently reveal important structure, particularly if the 3-D image if appropriately processed prior to surface extraction. Shaded displays of surfaces aid the understanding of the flow. A central problem with interpretation is the superposition of structures (hidden surfaces) when the surfaces are convoluted. We have addressed this problem with a surface peeling approach to dissecting the displayed surfaces.

While viewing the flow surfaces in stereo, the observer interactively segments the initially undifferentiated surfaces into a disjoint set of subsurfaces that encode his interpretation of the flow structure. For example, obscuring front surface regions can interactively be peeled away to reveal hidden interior structure. Figure 16 shows a stereo pair of a V-shaped streamer. Figure 17 shows the same streamer after part of the front surface is peeled off to show the tubular structure thought to be part of a counter-rotating vortex filament pair. Structures can be tested for connectivity, and significant structures such as interconnections can be highlighted. The mechanism is a sequence of 3-D connected surface voxel traversals, each starting at a cursor specified seed voxel and stopping according to various criteria, such as hitting a cursor drawn boundary, exiting a subvolume centered at the seed and/or exceeding a threshold angle between the local surface normal and the line of sight to the observer.

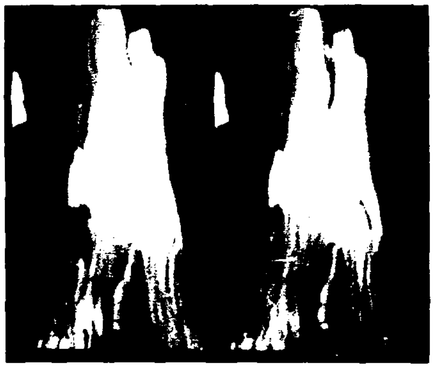


Figure 16: Stereo Pair of V-Shaped Streamer

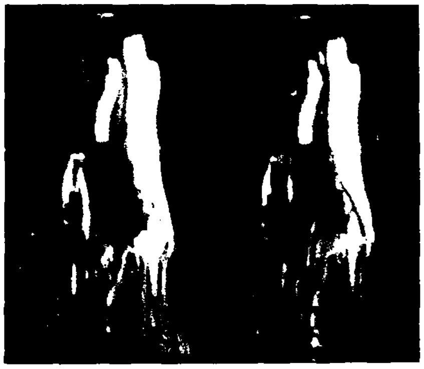


Figure 17: Dissection of streamer

Surface peeling is distinct from standard dissection methods such as cutting and contour level scrolling. Cutting can remove all surface regions within a specified subvolume. It is simple and useful, but does not reflect the specific structure of the surfaces. Surface peeling is more effective at preserving and focusing on interior interconnections because it selectively follows the flow structure. One can also generate a sequence of surfaces by varying the contour level. This approach can show how structure varies with intensity level, but is not suited to understanding connectivity relationships.

4. ACKNOWLEDGMENTS

We would like to thank other people who have also contributed to the preparation of their research efforts described in this paper. They are Steven Jaffey and Sylvia Shen. Bonnie Trang assisted in creating several of the Figures, and Dorothy Lohr provided efficient typing. The authors would also like to acknowledge the consistent support and guidance provided by Jim Pearson, Chuck Kuglin, Jerre McCulley, and Chuck Hall in the development of image processing and image understanding technology at Lockheed.

5. REFERENCES

1. C. M. Bjorklund and R. S. Loe, "Target Identification using Three-Dimensional Features", SPIE, Volume 267, 1982.
2. J. G. Landowski and R. S. Loe, "Object Detection using Region Growing in Laser Range Imagery", SPIE, Volume 938.
3. J. M. Barrilleaux and M. F. Doherty, "Multi-pass multi-resolution image interpretation", SPIE Conference on Image Understanding and the Man-Machine Interface II, 15-20 Jan 1989.
4. C. M. Bjorklund, Y. T. Liow, and T. Pavlidis, "Recognition of Urban Scenes", SPIE Conference on Image Understanding and the Man-Machine Interface II, 15-20 Jan 1989.
5. M. F. Doherty, "A Corner Detection Algorithm for Buildings in Visible Imagery", LMSC-D060626, December 1988.
6. M. F. Doherty, "Improved Cartographic Classification via Expert Systems", SPIE Appl. of Artif. Intel. III, Vol. 635, 1-3 Apr 1986, p. 35.
7. R. H. Laprade, "Split-and-Merge Segmentation of Aerial Photographs", Computer Vision, Graphics, and Image Processing vol. 44, pp. 77-86 (1988).
8. M.T. Noga and M. F. Doherty, "MIMD Image Segmentation", 1988 SPIE Workshop on Applied Imagery and Pattern Recognition, Nov. 2-4, Washington, D.C.
9. Jaffey, S.M., "Manipulating displayed surfaces of fluid flow structures", Invited paper, National Computer Graphics Association, Conference Proceedings Vol 3, Mar 1988c (Appendix E).
10. M. Jordan, "High resolution stereo parallax determination using a neural network," Proc. of SPIE OE/LASE , Los Angeles (1989).
11. K. Smedley, B. Haines, D. Van Vector, and M. Jordan, "Digital perspective generation and display of composite ocean bottom and coastal terrain images," Proc. of SPIE OE/LASE , Los Angeles (1989).
12. W. E. L. Grimson, "Computational Experiments with a Feature Based Stereo Algorithm", IEEE Trans. PAMI, vol. PAMI-7, pp. 27-34, January, 1985.
13. G. Medioni and Nevatia, R., "Segment-Based Stereo Matching", Computer Vision, Graphics, and image Processing, vol. 31, pp. 2-18, July, 1985.
14. S. S. Shen, "A multi-resolution Approach to Image Segmentation", Proceedings of SPIE on Applications of Digital Image Processing, vol. 829, pp. 65-70, 1986.
15. E. Barrett, and P. Payton, "Projective Invarients in Imagery", 1988 Conference on Pattern Recognition for Advanced Missile Systems, Nov. 14-15, Redstone Arsenal, Huntsvilee, AL.
16. E. Barrett, and D. Mighell, "Applications of Neural Networks to Linear Programming, Continueous Logic, and Evidentiary Reasoning", in preparation.
17. C. Smymiotis, P. Payton, and E. Barrett, "A Knowledge-Based Imagery Exploitation System", 1988 SPIE Workshop on Applied Imagery and Pattern Recognition, Nov. 2-4, Washington, D.C.
18. T. Pavlidis, "Image Analysis," Ann. Rev. Comput. Sci. 3, 121-146 (1988).
19. S. Sinton and S.L. Huff, "Aggregation and Interfacial Properties of an Isomerically Pure Alkylaryl Sulfonate," J. Colloid and Interface Sci. 120, 358 (1987).
20. M. Chen, T. Pavlidis, and M. Noga, "Tiling strategies for image parallelism," Proc. of SPIE OE/LASE , Los Angeles (1989).
21. A. Title, et. al., "Large-scale horizontal flows from SOUP observations of solar granulation," Theoretical Problems in High-Resolution Solar Physics II, NASA CP-2483, 55 (1987).
22. D. Kuan, H. Shariat, K. Dutta, and P. Ransil, "A constraint-based system for interpretation of aerial imagery," Proc. of the 2nd Intl. Vision Conf. on Comp. Vision, Tampa, FL, 601-609 (1988).

GEOMETRIC PROBLEM SOLVING BY MACHINE VISUALIZATION

Extended Abstract

Rand Waltzman

Computer Vision Laboratory
Center for Automation Research
University of Maryland
College Park, MD 20742-3411

Visualization, in general, and the direct use of images and diagrams, in particular, has long been recognized as a powerful technique in human problem solving. In this talk I will present my first results in an investigation of how images and diagrams can be used in problem solving by a machine.

A central theme of this work is that a powerful technique for reasoning in a problem domain is to embed the semantics of the domain in a representation system so that syntactic operations correspond to domain operations. I have exploited this idea in the domain of geometry. The results of this work demonstrate that powerful "non-logical" syntactic techniques can be used for reasoning in domains such as geometry where traditional logic-based techniques have had limited success.

As a vehicle for this investigation I have built a system for solving problems of the following type: We are given a set of three-dimensional polyhedral pieces and told that they all fit together to form a solid cube (i.e., no empty space inside) of given dimensions. The problem is to determine how all the pieces fit together. This problem can also be thought of as a packing problem, i.e., how can the set of pieces be packed into a cubic container of the given dimensions. The individual objects in this problem are the polyhedral pieces. Geometrical properties of interest might be symmetries, concavities, number of faces, dimensions, etc. Interesting geometrical relations between polyhedra might be lock-and-key relations (does one polyhedron fit into another), contact (e.g., which faces are touching), relative orientations, etc. A problem-solving heuristic that we might use is: Always place pieces in such a way that a vertex of a piece exactly fits into a vertex of the container.

More formally, the problem can be stated: Find the spatial location and orientation of each piece relative to a cubic container of a given dimension so that the container is completely filled by the pieces. Each solution step consists of placing a piece at a particular location and orientation in space. The basic problem state at any given time consists of the locations and orientations of all the pieces placed so far, as well as the shapes, locations, and orientations of the containers remaining to be filled. (Note that at any stage there may be more than one container to be filled since the placement of any piece may divide a single container into two or more containers.)

The problem that I have described is an important component of several well known problems for which completely satisfactory solutions have not yet been found. There is the general class of problems of packing a set of objects into a container. The container might be a means of transportation (e.g., ship, plane, or truck) or of storage (e.g., shelves or bins). Related to this are problems of laying out office space or factory floor space where the container is the total floor space that we have to work with and the objects are wall partitions, desks, office equipment, and factory equipment. Automatic assembly planning problems form another important class. Knowledge about automatic assembly planning can be incorporated into computer aided design (CAD) systems where, for example, a critiquing component of the CAD system could interactively provide advice to the designer about the manufacturability of a given design.

In many of these problems, the required packing or arrangement of objects must satisfy conditions other than the one that the objects geometrically fit. For example, when loading cargo onto an airplane, there are certain requirements about the distribution of weight that must be satisfied. When arranging machinery on a factory floor, there must be enough space left to allow access for maintenance. When planning an assembly operation, consideration must be given to the capabilities of the mechanical manipulators to be used. I have not addressed packing problems under these types of constraints in this work. The examples are restricted to purely geometric packing. However, the techniques developed here represent a significant step in the solution to these more practical problems.

The goal of this work is to develop a framework for dealing with a limited type of geometric problem solving, to identify some of the basic issues involved, and to suggest concrete ways of addressing some of these issues in the context of constructing a system to solve problems like the example problem.

The framework for geometric problem solving that I propose consists of the following two parts:

- a spatial algebra
- a geometrically salient formalism for knowledge representation

A SPATIAL ALGEBRA

A powerful feature of logic as a representation language is that it provides a general technique for expressing the semantics of a problem domain through axiomatization. Then, objects in the problem domain can be manipulated and reasoned about using purely syntactic means. For example, automatic theorem provers can prove semantically meaningful results in a given domain using domain-independent resolution techniques. Though this use of logic has been shown to be effective for many domains, success in geometry has been limited. This is due, in part, to the fact that generality is achieved at the cost of the ability to explicitly represent important domain-specific structural information. Though this information is implicitly available, i.e., could be computed, the lack of immediate accessibility can be the source of tremendous inefficiency in the problem solving process. The question is whether there is a representation for geometric objects, not based on logic, that would allow more explicit representation of this domain-specific structural information and admit syntactic operations that directly correspond to meaningful geometric manipulations. I have shown that the answer to this question is positive by describing such a representation. I will refer to the set of corresponding syntactic operations as the spatial algebra of the representation.

INTRINSIC AND ISOMORPHIC REPRESENTATION

Most of the existing geometric problem solving systems are logic based and function by extracting geometric properties and relations that can be represented by high-level predicates from a low-level representation of the geometrical objects involved. This low-level representation is generally in the form of an image stored as an array of pixels or as a set of point coordinates. By using predicates in this way the immediate geometry is lost to the problem solving system. My proposed knowledge representation scheme operates at an intermediate level; an enhanced version of the representation used to describe the geometric objects themselves is used to represent problem solving knowledge. This geometrically salient formalism allows problem-solving knowledge to be expressed directly in terms of the structure of the objects that are the subject of the reasoning process.

The geometric representation proposed here is intrinsic. It is coordinate free and yet contains complete metrical and topological information for the objects represented, i.e., objects are completely described independent of location and orientation in space. The representation is also unique, i.e., there is a one to one relation between representation and object. Finally, the representation is isomorphic in the sense that the structure of the representation directly reflects the structure of the objects being represented. This is an example of what I call a visual data structure. The nature of visual data structures makes them well suited for use in determining structural characteristics of individual objects (e.g., symmetries) as well as relations between structural characteristics of distinct objects.

The system tries to detect patterns in the geometrical structure of the current problem state that suggest the next best move (e.g., which piece to place and where to place it) at each step in the problem-solving process. A major goal of this work is to develop the capability of representing and using problem-solving knowledge which allows the system to exploit geometrical structure that the problem might have. For example, if there are symmetries in the pieces or the container into which the pieces are to be placed, the system could take advantage of them in deciding the most effective move. A problem-solving heuristic in this case might be: Place pieces which are mirror images of each other into container locations which are equivalent under a mirror symmetry of the container.

In addition, there will be a variety of geometric computations (e.g., compute the [illegible] ne current spaces to be filled, determine whether a given piece will fit into the space at a given location, etc.) that may be relevant at each step in the problem-solving process. Some of these computations could contribute to or even provide solutions to restricted versions of more general problems. For example, one might ask whether given two polygons there is any way that one of them might fit inside the other. A more restricted version of this problem would be: Given two polygons A and B, will A fit inside B given that vertex a of A coincides with vertex b of B, the size of a is the same as that b, and the two sides of A that form a are shorter than the corresponding sides of B that form b? The second problem is much easier to solve than the first. Part of the problem-solving strategy of the system is to use algorithms which solve such restricted versions of more general problems by trying to guess when their application will produce useful information, i.e., to use them heuristically. For this scheme to be worthwhile, it must be relatively easy for the system to recognize the appropriate restrictive geometric conditions in a problem under which a given algorithm is applicable. I will show how this can be done using the proposed representation.

PROBABILITY-BASED CONTROL FOR COMPUTER VISION

Tod S. Levitt[1], Thomas O. Binford[2], Gil J. Ettinger[1], Patrice Gelband[1]

[1] Advanced Decision Systems, 1500 Plymouth Street, Mountain View, CA
[2] Stanford University, Stanford, California

1. Introduction

Several key issues arise in implementing recognition in terms of Bayesian networks. Perceptual networks are very deep, typically fifteen levels of structure. For efficiency, we dynamically instantiate hypotheses of observed objects. The network is not fixed, it is only partially instantiated. Hypothesis generation and indexing are important, but they are not considered here [Nevatia - 73], [Ettinger - 88]. This work is aimed at near-term implementation with parallel computation in a radar surveillance system, ADRIES [Levitt - 88], [Franklin - 88] and a system for industrial part recognition, SUCCESSOR [Binford - 88].

For many applications, vision must be faster to be practical and so efficiently controlling the machine vision process is critical. Perceptual operators scan megapixels and may require minutes of computation time. It is necessary to avoid unnecessary sensor actions and computation. The potential for parallel, distributed computation for high-level vision means distributing non-homogeneous computations. This paper addresses the problem of control in machine vision systems based on Bayesian probability models.

We separate control and inference to extend the previous work [Binford - 89] to maximize utility instead of probability. Maximizing utility allows adopting perceptual strategies for efficient information gathering with sensors and analysis of sensor data. Results of controlling machine vision via utility to recognize military situations are presented in this paper. Future work extends this to industrial part recognition for SUCCESSOR.

2. BAYESIAN NETWORK FOR EVIDENTIAL ACCRUAL

The relationship between models, hypotheses, and decisions is pictured in Figure 1. Models represent physical objects in the world, such as military units, formations, industrial parts, components of parts, and attributes such as color, reflectivity, etc. As such, we view our models as causal; i.e., a physical object is viewed as "causing" its component sub-parts.

Object models are physical models; their geometry is represented by part/whole graphs and by interlocking taxonomic graphs. Figure 2 shows two part-of slices of the (taxonomic) is-a hierarchy for a model of a military brigade of the evil empire of Mordor. The part-of hierarchy corresponds to (physical) military sub-units. The is-a hierarchy is obtained by taking the common set of unit type and formation constraints for military units that can be confused based on uncertain observations. For example, if we are too far away to distinguish steam engines from catapults, we might still recognize them as vehicles, and be uncertain as to whether we are observing a Catapult Battalion or a Steam Engine Team. For military units, the models of military organization predict the existence and location of other sub-units, given the observation of another.

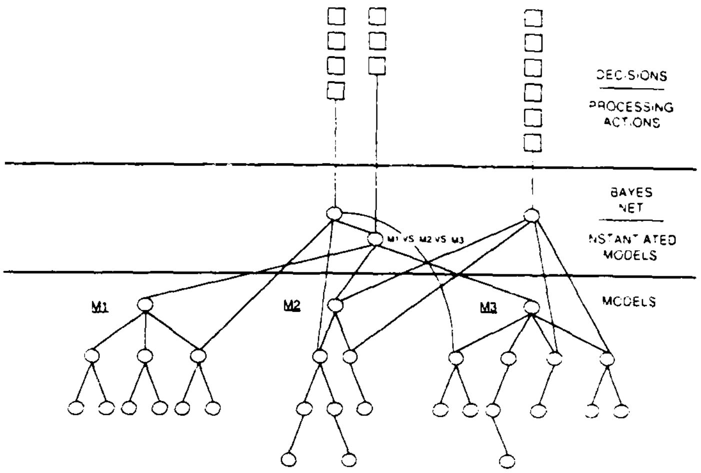
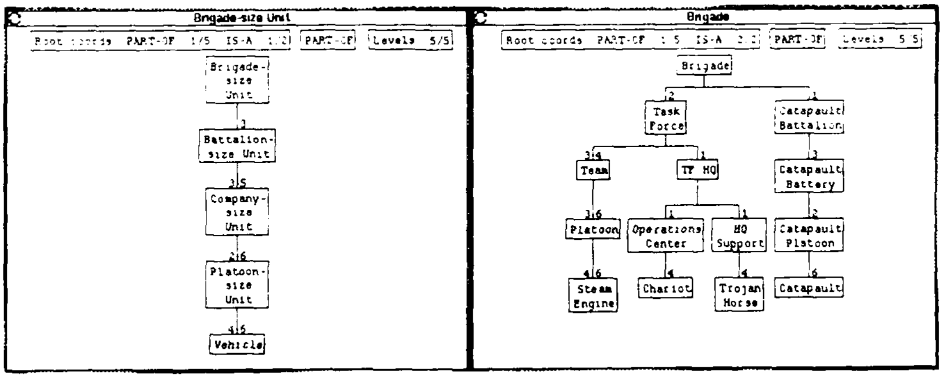


Figure 1: Model-Hypothesis-Decision Relationships



Brigade-size Unit

Root coords PART-OF 1/5 IS-A 1/2 | PART-OF | Levels 5/5

Brigade

Root coords PART-OF 1/5 IS-A 2/2 | PART-OF | Levels 5/5

Figure 2: Brigade Model

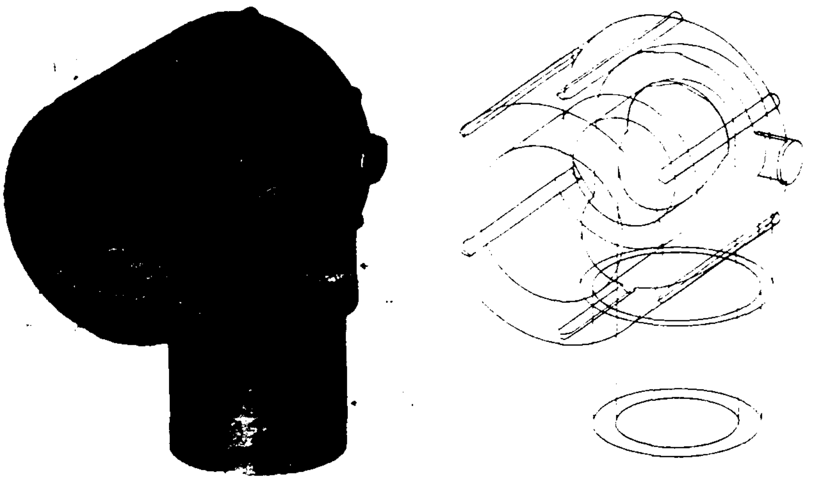


Figure 3: a. Elbow without Threads b. Line Drawing of Elbow without Threads

In optical part recognition for manufacturing, we represent objects as part-of hierarchies based on generalized cylinder volume primitives. Object models are recursively broken up into joints composed of parts; those parts may in turn be broken into sub-joints and sub-parts, or they may be primitive. Joints are relationships between parts, incorporating observable effects of joining parts. Such a hierarchy forms a directed acyclic graph (DAG), where nodes are parts or relations and arcs indicate part-of relationships. Generalized cylinders (GCs) are defined by a cross section swept along a space curve, the axis, under a sweeping transformation [Ponce - 88]. Compound object models are DAGS of primitives represented in a simple modelling language. Models also include material modeling of optical properties, i.e., reflectives, specularities, and color [Healey - 87, 88]. Figure 3a shows an elbow without threads. Figure 3b shows the line drawing of the elbow without hidden line suppression to show its subparts.

At each node of the Bayes net, there is a probability distribution over the set of mutually exclusive and exhaustive possible interpretations of the visual evidence accrued to that level in the hierarchy. A node is a set of *hypotheses*, e.g., *catapult-battalion* vs. task-force vs. non-military-unit, or t-joint versus elbow-joint versus non-joint. Although they do not have to be simultaneously instantiated, the possible links between nodes are hard-wired, a priori, by the models of objects and relationships, and the criteria for node *instantiation that determine which pieces of evidence can generate conflicting* hypotheses. Each alternative hypothesis at a node contributes some probability to the truth of an alternative hypothesis at a parent node (e.g., the part supports the existence of the whole) and also contributes to the truth of supporting children. When new evidence appears at a node, it is assimilated and appropriate versions of that evidence are propagated along all other links entering or exiting the node. We use the propagation algorithms of Pearl [Pearl - 86], [Binford - 87].

As we dynamically create the Bayes net at runtime, node instantiation is guided by the a priori models of objects, the evidence of their components, and their relationships. System control alternates between examination of the instantiated Bayes nodes, comparing against the models, and choosing what actions to

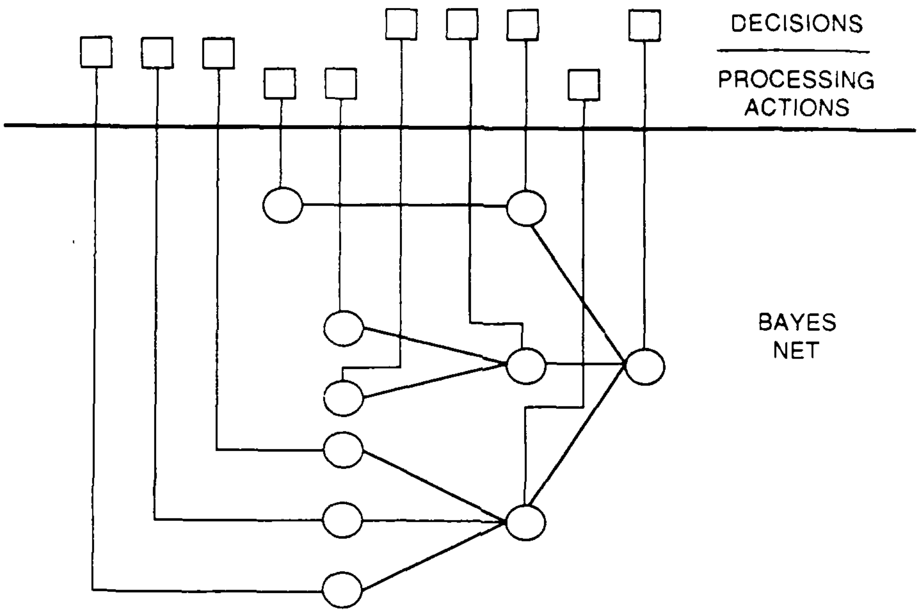


Figure 4: Separable Influence Diagram

take to grow the net, which is equivalent to seeing more structure in the world. Thus, inference proceeds by choosing actions from the model space that create new nodes and arcs in the Bayes net. All possible chains of inference that the system *can perform* are specified *a priori* in the model-base. This feature clearly distinguishes inference from control. Control chooses actions and allocates them over available processors, and returns results to the inference. Inference uses the existing Bayes net, the current results of actions (i.e., the collected evidence) generates Bayes nodes and arcs, propagates probabilities over the net, and accumulates the selectable actions for examination by control. In this approach, it is impossible for the system to reason circularly, as all instantiated chains of inference must be supported by evidence in a manner consistent with the model-base.

The prioritization and selection of actions can be viewed as a decision-making procedure. By representing the selection of actions at a single Bayes-node as a single decision, we create an influence diagram [Shachter - 86] with the property that severing any decision node from the diagram leaves the Bayes-net intact. Figure 4 illustrates this design. This allows us to construct control algorithms over the influence diagram where evidence accrual in the Bayes-net, and decisions of actions to execute, appear as modular operations.

3. UTILITY FOR EVIDENCE-GATHERING ACTIONS

Our approach to selecting actions by utility theory is to compute the estimated value and cost of each action, then maximize value constrained by a bound on the total cost. We define cost of an action as the average processing time for the action. If the action is an algorithm that can be performed on different processors with radically different computation times, we can model this as two different actions.

The computation of value is performed hierarchically over the Bayes net, where hierarchy is the hierarchy inherent in the model space. That is, we view computing the value of an action, A, at the child-hypothesis, H_k, as the increment in evidential value achieved at the parent hypothesis, P. We define the value, V, as

$$\begin{aligned} V(\text{Child, Action}) &= V(H_k, A) \\ &= \sum_{\text{Parents}} [p(\text{Parent}\,|H_k, A) - p(\text{Parent}\,|H_k)] \cdot V(\text{Parent}) \end{aligned}$$

Thus, we can begin at the top level of the model hierarchy and assign values to recognizing, for example, the various military units or industrial parts. We then, recursively compute the value at each child, or sub-part, down the model hierarchy. In the instantiated Bayes-net, this computation is proportional to the number of instantiated levels in the hierarchy. For example, in trying to confirm or deny the presence of a task force, we can assign a value of .8 to H_1 = task force, .1 to H_2 = catapult battalion and .1 to H_3 = other. These are the objects in the goal Bayes node. The actions include "search for sister sub-unit", "get closer observation of vehicle types", and "adjust match of formation based on adaptation to underlying terrain".

If we have a set of child hypotheses, H_k, at Bayes-node N, then the value of taking action A at node N is defined as

$$V(N, A) = \sum_k V\,(H_k, A)$$

If action A has cost t_A, we maximize expected value

$$\sum_{(N,A)} \chi_A \, V(N, A)$$

subject to the constraint that total cost is bounded by T:

$$\sum_A \chi_A \, t_A \leq T$$

where

$$\chi_A = \begin{cases} 1 \text{ if we perform A} \\ 0 \text{ if we do not perform A} \end{cases}$$

and T is the maximum allowable processing time. For any fixed T, we produce an equivalence class of plans of actions to be performed, and the results of executing these plans are recognized objects with probabilities.

We vary T to obtain the desired level of performance, i.e., we *generate sets of plans for each value of* T. We typically choose the minimum T for a desired probability of recognition.

There is an implicit assumption in this approach that all executable actions are represented in the model space a priori, and that values are calculated to account for continuous ranges of values for individual pieces of evidence. For example, executing a procedure to infer the curvature of a part may depend upon hypothesizing and testing against possible curvatures. We use the expected value of a quasi-invariant measure given its observation as in [Binford - 87].

We then use an integer optimization procedure to select over the possible sets of actions executable in the allowed time. The required algorithm for maximizing utility must solve the classic "knapsack" problem. The knapsack problem is to

$$\text{maximize} \sum_i V_i x_i$$

$$\text{subject to the constraints} \sum_i x_i t_i \leq \text{T and } x_i = 0 \text{ or } 1.$$

The x_i are evidence gathering actions, V_i is the value of a given action, and t_i is the time to perform the action. So *the problem is to maximize* value by choosing which actions to perform (corresponding to $x_i = 1$) and which not to perform (corresponding to $x_i = 0$) while staying within the time limit.

The knapsack problem is an NP hard problem [Garey - 79]. Because we expect to be dealing with on the order of 100 actions, it is infeasible to solve the problem exactly. Therefore, we use an algorithm that finds an approximate solution. Specify the desired accuracy ϵ, and it finds a solution satisfying $\frac{(\mathrm{P}' - \mathrm{P})}{\mathrm{P}'} \leq \epsilon$, where P' is the total value of the optimal solution. The algorithm has time complexity $O\ (\mathrm{N} \ln \mathrm{N}) + O\left(\left(\frac{3}{\epsilon}\right)^{2 \bullet \mathrm{N}}\right)$ and space complexity $O(\mathrm{N}) + O\left(\left(\frac{3}{\epsilon}\right)^{3}\right)$ where N is the number of actions.

We now return to computing

$$p\,(\text{Parent}\,|\,\text{Child, Action}).$$

In general, the increase in belief in the parent depends on the results of computations performed in the action, which can, in turn, depend on many other results of processing at other nodes corresponding to sub and super-hypotheses. We apply Bayes rule.

$$p\,(\text{Parent}\,|\,\text{Child, Action}) = \frac{p\,(\text{Child, Action}\,|\,\text{Parent})\; p\,(\text{Parent})}{p\,(\text{Child, Action})}$$

and note that p (Parent) is known at runtime. When a Bayes node is already instantiated, and the p (Child, Action) can be interpreted as the accuracy with which the results of the action can be measured, given the state of the child. For example, if the child is a boundary of a generalized cylinder of the parent and the action is a curvature measurement, then the joint probability can determine how accurately the curvature can be measured, given the pixels observed on the boundary.

Finally, the term p(Child, Action |Parent) is defined as

$$\int_{\text{outcomes of action}} p(\text{Child, Outcome}\,|\text{Parent})$$

where p(Child, Outcome |Parent) is computed and stored a priori. For example, if the child is a pair of generalized cylinders, the action is an angular measurement between them, and the parent is a joint with known angular measure; then the above formula specifies the probability we would observe a given outcome (angle) given the true (model) angle. See [Binford - 87] for an example of such a computation.

Now if a higher level Bayes node, e.g., a generalized cylinder, is not yet instantiated, but we wish to compute the value of actions at an instantiated lower level node, e.g., an observed edge of a generalized cylinder, then the probability, p(generalized cylinder), must be estimated a priori for the recursive computation of value at the observed edge node. We take these priors to be the task-based likelihood that given objects are present in a scenario. For example, in an assembly line application, based on the current manufacturing task, we have an a priori notion of what parts to expect on the line.

Control in the influence diagram is effected by the top-level loop of: execute actions, accrue probabilities, compute values, maximize utility, and select actions. A version of this algorithm in terms of the Bayes-net is given in Figure 5.

Execution of actions can occur on multiple machines in a distributed environment. Results are summarized and returned asynchronously to the Bayes net. We have structured the model space, and therefore the Bayes net, such that the assumptions of Pearl's algorithm [Pearl - 86] are fulfilled. This allows asynchronous updating and propagation of probabilities, throughout the net. Because no decision nodes are between Bayes nodes, Pearl's algorithm applies over the subsets of the influence diagram that are connected Bayes-nets. Note that this structuring of the influence diagram, see Figure 4, was necessary to permit a control structure in which probability accrual, and decision making are separable operations.

4. EXAMPLE

The following example presents the use of utility-based control to drive the recognition of military units from aerial imagery. The aerial imagery used is assumed to be relatively low resolution so that individual vehicles are difficult to identify due to their small size and a high false alarm rate. As a result additional contexual forms of evidence are used to recognize the military forces. The acquired evidence is matched against known military force models in order to determine its support. The force models resident in the system are shown in Figure 2. The recognition system normally commences processing by generating hypotheses for the coarse models and proceeds by refining them and using them to generate higher-level hypotheses. The Bayes net is

```
Until ((time exceeded or (termination condition achieved))
      For each instantiated Bayes node
            Get list of possible actions from node
            Evaluate value of each action
      End for
  Until (all processors allocated or all actions selected)
      Maximize expected value constrained by total processing time
      Allocate selected actions over available processors
  End Until
      Until ((a node's probability ratio exceed's threshold)
            or (all k actions return values))
            propagate evidential returns over Bayes net
            update values at node
      End Until
End Until
```

Figure 5: Control Algorithm

then used to group conflicting hypothesis configurations and to propagate beliefs throughout the hypothesis space.

In this example we are attempting to confirm the presence of a Brigade in the boxed region in the upper portion of the map shown in Figure 6a. The system is initialized by locating possible vehicle detections in the available imagery. These detections are then clustered into Company-size Unit hypotheses based on coarse parameters in the model database such as inter-vehicle distance, number of vehicles in a unit, and maximum extent of a unit. By initializing the Bayes net with Company-size Units and not individual vehicles we achieve a large reduction in combinatorics. The initial cluster units are shown in Figure 6b.

After initialization the system progresses by performing any of the following actions on the appropriate Bayes nodes:

- Refine a Bayes net hierarchy by using the more detailed force type model description (Refine-type).
- Refine a Bayes net hierarchy by using a more detailed *formation description* (Refine-*formation*).
- Search for matches among lower-level force hypotheses in order to generate higher-level force hypotheses (Search).
- Attach terrain evidence to a Bayes node by examining the support the underlying terrain provides for the given force (Terrain-support).
- Attach classification evidence to a Bayes leaf node indicating the support for the given force type obtained from high resolution sensors an accurate process that is normally expensive to perform (Classification).

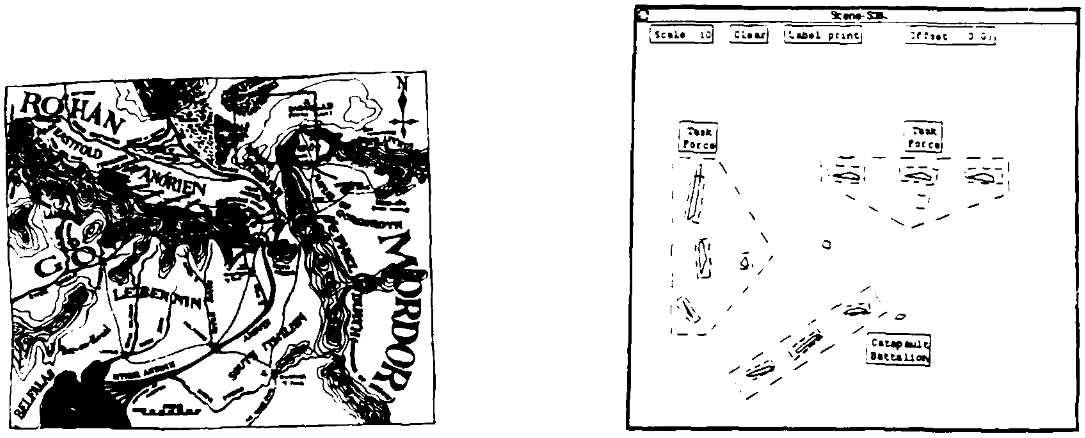


Figure 6: a. Area of Interest b. Initial Hypotheses (Solid Lines) and Actual Ground Truth (Dashed Lines)

At each step, the system generates the available actions and computes their utility based on value and cost models derived from previous system performance. Optimal actions are then selected by maximizing the expected value of the actions that can be executed in a given time step using the knapsack approximation algorithm. Table 1 lists the actions selected by the system at each step.

The initial Bayes net after generation of Company-size Unit hypotheses is shown in Figure 7a. The attached probabilities are the ones obtained from the detection likelihoods and clustering matches. After the actions in step 1 were performed, the Bayes net contained refined hypotheses for Team, Task Force Headquarters, and Catapault Battery. These refined hypotheses are shown in Figure 7b. The next iteration executed actions supplying terrain and classification support for the four Bayes nodes. The resultant probabilities are shown in Figure 8a. The hypotheses with high belief are depicted graphically in Figure root-3b. The third iteration step searched for matches among the likely hypotheses and generated Task Force and Catapault Battalion hypotheses. These are shown in Figures 9a and 9b. The fourth step directed the system to match the Task Force and Catapault Battalion hypotheses into Brigade hypotheses. The resultant Bayes net is shown in Figure 10 and the locations of the two resultant hypotheses are shown in Figures 11a and 11b. The fifth step attached terrain support to the Brigade hypotheses, a process that resulted in the Brigade hypotheses receiving high support (Figure 12). As a result the system reported that a Brigade most likely exists in the area (since the two hypotheses conflict, the belief that a Brigade is present is .99) and its exact location is given by the likelier hypothesis.

5. CONCLUSIONS

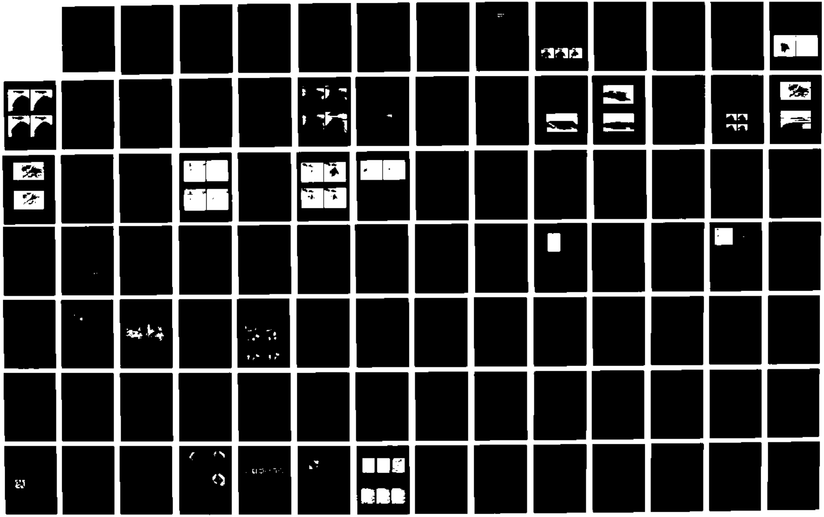
We have developed a methodology for vision system control based on utility theory applied to model-based Bayesian inference. We have implemented this methodology in ADRIES, a radar surveillance system, and are implementing it in SUCCESSOR, a system for computer vision of industrial parts in optical imagery.

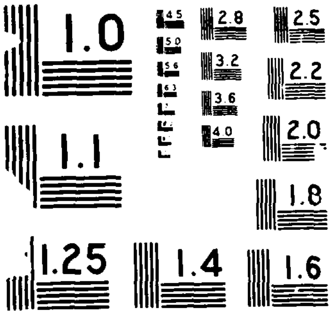
Table 1: Actions Selected by the System at Each Step

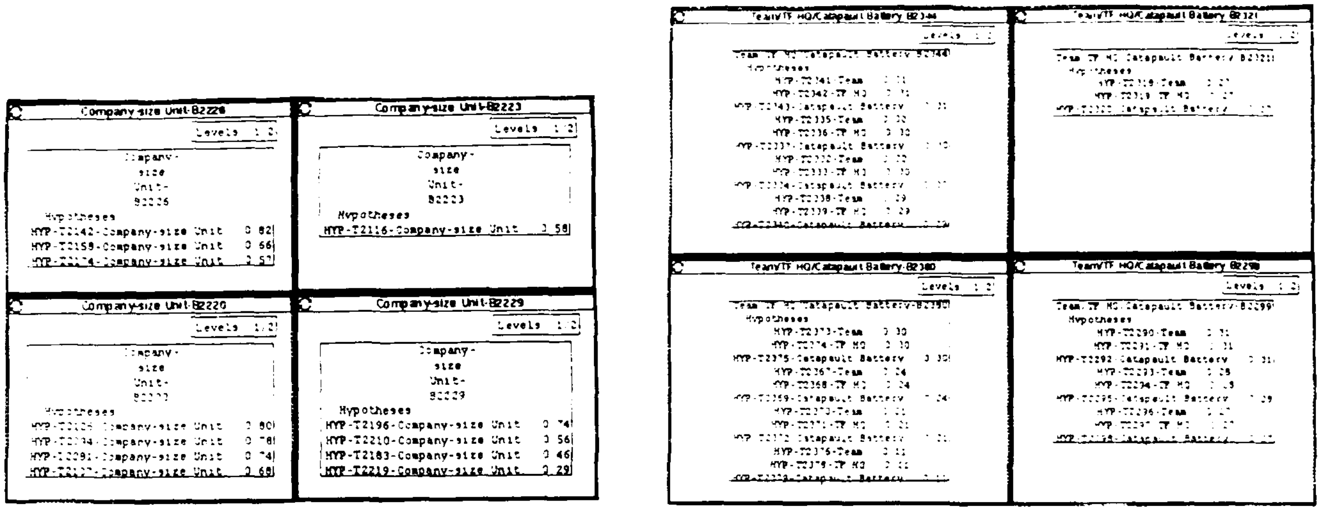
Step	Selected Actions	Value	Cost
1	(REFINE-TYPE of(<B2226> <B2223> <B2220> <B2229>))	11522	1600
	(SEARCH for matches of hypotheses in (<B2226> <B2223> <B2220> <B2229>))	5761	842
	(TERRAIN-SUPPORT for <B2220>)	1125	820
	(TERRAIN-SUPPORT for <B2226>)	769	820
	(TERRAIN-SUPPORT for <B2229>)	769	820
	(TERRAIN-SUPPORT for <B2223>)	217	820
2	(CLASSIFICATION-SUPPORT for <B2344>)	619	1320
	(TERRAIN-SUPPORT for <B2344>)	326	820
	(CLASSIFICATION-SUPPORT for <B2299>)	445	1320
	(CLASSIFICATION-SUPPORT for <B2380>)	441	1320
	(TERRAIN-SUPPORT for <B2299>)	234	820
	(TERRAIN-SUPPORT for <B2380>)	232	820
	(CLASSIFICATION-SUPPORT for <B2321>)	139	1320
	(TERRAIN-SUPPORT for <B2321>)	73	820
3	(SEARCH for matches of hypotheses in (<B2344> <B2321> <B2380> <B2299>))	2359	841
4	(SEARCH for matches of hypotheses in (<B2739>))	1672	832
	(REFINE-FORMATION of <B2739>)	1060	8100
	(SEARCH for matches of hypotheses in (<B2321>))	20	801
5	(TERRAIN-SUPPORT for <B2739>)	281	2300
	(TERRAIN-SUPPORT for <B3230>)	252	2300

AD-A212 981 PROCEEDINGS OF IMAGE UNDERSTANDING WORKSHOP HELD IN 5/13
PALO ALTO CALIFORNIA O (U) SCIENCE APPLICATIONS
INTERNATIONAL CORP MCLEAN VA L S BAUMAN MAY 89

UNCLASSIFIED N00014-86-C-0700 F/G 17/11 NL




1.0
4.5
5.0
5.6
6.3
2.8
3.2
3.6
4.0
2.5
2.2
2.0
1.1
1.8
1.25
1.4
1.6



```
Company-size Unit-B2226
                                Levels 1/2
        Company-
        size
        Unit-
        B2226
  Hypotheses
HYP-T2142-Company-size Unit    0.82
HYP-T2159-Company-size Unit    0.66
HYP-T2174-Company-size Unit    0.57

Company-size Unit-B2223
                                Levels 1/2
        Company-
        size
        Unit-
        B2223
  Hypotheses
HYP-T2116-Company-size Unit    0.58

Company-size Unit-B2220
                                Levels 1/2
        Company-
        size
        Unit-
        B2220
  Hypotheses
HYP-T2125-Company-size Unit    0.80
HYP-T2094-Company-size Unit    0.78
HYP-T2091-Company-size Unit    0.74
HYP-T2127-Company-size Unit    0.68

Company-size Unit-B2229
                                Levels 1/2
        Company-
        size
        Unit-
        B2229
  Hypotheses
HYP-T2196-Company-size Unit    0.74
HYP-T2210-Company-size Unit    0.56
HYP-T2183-Company-size Unit    0.46
HYP-T2219-Company-size Unit    0.29
```

```
Team/TF HQ/Catapault Battery-B2344
                                Levels 1/2
Team/TF HQ/Catapault Battery-B2344
  Hypotheses
      HYP-T2341-Team    0.31
      HYP-T2342-TF HQ    0.31
HYP-T2343-Catapault Battery    0.31
      HYP-T2335-Team    0.30
      HYP-T2336-TF HQ    0.30
HYP-T2337-Catapault Battery    0.30
      HYP-T2332-Team    0.30
      HYP-T2333-TF HQ    0.30
HYP-T2334-Catapault Battery    0.30
      HYP-T2338-Team    0.29
      HYP-T2339-TF HQ    0.29
HYP-T2340-Catapault Battery    0.29

Team/TF HQ/Catapault Battery-B2321
                                Levels 1/2
Team/TF HQ/Catapault Battery-B2321
  Hypotheses
      HYP-T2318-Team    0.27
      HYP-T2319-TF HQ    0.27
HYP-T2320-Catapault Battery    0.27

Team/TF HQ/Catapault Battery-B2380
                                Levels 1/2
Team/TF HQ/Catapault Battery-B2380
  Hypotheses
      HYP-T2373-Team    0.30
      HYP-T2374-TF HQ    0.30
HYP-T2375-Catapault Battery    0.30
      HYP-T2367-Team    0.24
      HYP-T2368-TF HQ    0.24
HYP-T2369-Catapault Battery    0.24
      HYP-T2370-Team    0.21
      HYP-T2371-TF HQ    0.21
HYP-T2372-Catapault Battery    0.21
      HYP-T2376-Team    0.11
      HYP-T2379-TF HQ    0.11
HYP-T2378-Catapault Battery    0.11

Team/TF HQ/Catapault Battery-B2299
                                Levels 1/2
Team/TF HQ/Catapault Battery-B2299
  Hypotheses
      HYP-T2290-Team    0.31
      HYP-T2291-TF HQ    0.31
HYP-T2292-Catapault Battery    0.31
      HYP-T2293-Team    0.29
      HYP-T2294-TF HQ    0.29
HYP-T2295-Catapault Battery    0.29
      HYP-T2296-Team    0.27
      HYP-T2297-TF HQ    0.27
HYP-T2298-Catapault Battery    0.27
```

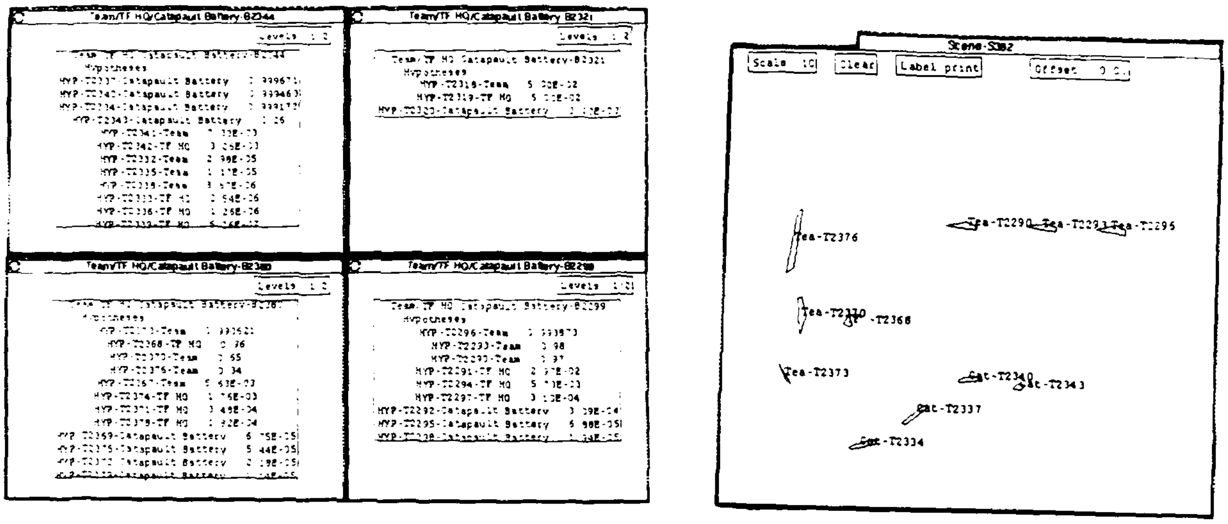
Figure 7: a. Root Nodes of the Initialized Bayes Net b. Root Nodes of the Bayes Net after the First Step

Many technical innovations were developed including:

- Representation of control and inference in a cognitively tractable model promoting clean and efficient system designs.
- Separation of decision making from evidence accrual.
- Dynamic instantiation of Bayes nets and influence diagrams.
- Hierarchical value computation achieved by assigning values only at the top model-level.
- Handling real world problems.

Acknowledgements

This work was supported by the Advanced Digital Radar Imagery Exploitation Systems (ADRIES) project sponsored by the Defense Advanced Research Projects Agency (DARPA) and the U.S. Army Engineer Topographic Laboratories (USAETL) under U.S. Government Contract No. DACA76-86-C-0010, and by the Knowledge-Based Vision Techniques (KBV) project sponsored by USAETL and DARPA under U.S. Government Contract No. DACA76-85-C-0005.



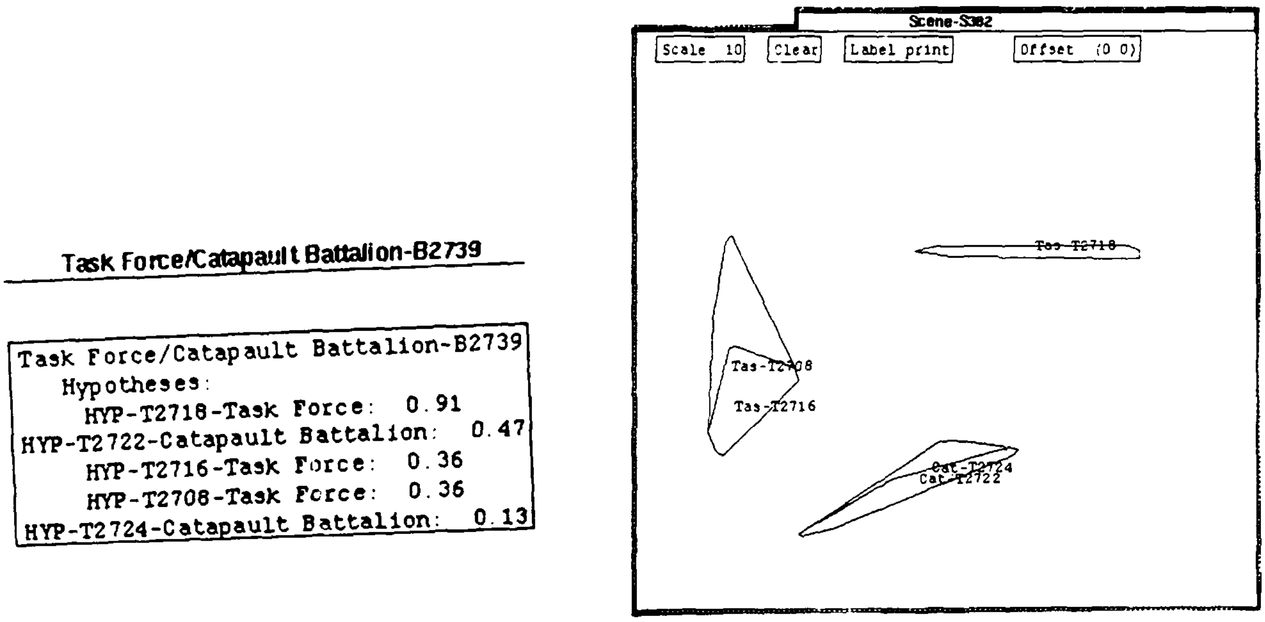
```
Team/TF HQ/Catapault Battery-B2344
  Hypotheses
HYP-T2337-Catapault Battery   0 999671
HYP-T2340-Catapault Battery   0 999463
HYP-T2334-Catapault Battery   0 999172
  HYP-T2343-Catapault Battery   0 26
      HYP-T2341-Team   7 33E-03
      HYP-T2342-TF HQ   3 26E-03
      HYP-T2332-Team   2 98E-05
      HYP-T2335-Team   1 17E-05
      HYP-T2339-Team   3 57E-06
      HYP-T2333-TF HQ   2 54E-06
      HYP-T2336-TF HQ   1 26E-06
      HYP-T233[illegible]-TF HQ   5 [illegible]
```

```
Team/TF HQ/Catapault Battery-B2321
  Hypotheses
    HYP-T2318-Team   5 00E-02
    HYP-T2319-TF HQ   5 00E-02
HYP-T2320-Catapault Battery   [illegible]
```

```
Team/TF HQ/Catapault Battery-B2380
  Hypotheses
    HYP-T2373-Team   0 999521
      HYP-T2368-TF HQ   0 76
      HYP-T2370-Team   0 65
      HYP-T2376-Team   0 34
    HYP-T2367-Team   5 63E-03
    HYP-T2374-TF HQ   1 76E-03
    HYP-T2371-TF HQ   3 49E-04
    HYP-T2379-TF HQ   1 32E-04
HYP-T2369-Catapault Battery   6 75E-05
HYP-T2375-Catapault Battery   5 44E-05
HYP-T2372-Catapault Battery   2 19E-05
HYP-T2377-Catapault Battery   [illegible]
```

```
Team/TF HQ/Catapault Battery-B2299
  Hypotheses
    HYP-T2296-Team   0 999973
      HYP-T2293-Team   0 98
      HYP-T2290-Team   0 97
    HYP-T2291-TF HQ   2 97E-02
    HYP-T2294-TF HQ   5 70E-03
    HYP-T2297-TF HQ   3 10E-04
HYP-T2292-Catapault Battery   3 39E-04
HYP-T2295-Catapault Battery   6 98E-05
HYP-T2298-Catapault Battery   1 34E-05
```

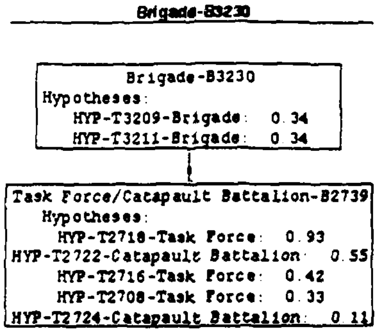
Figure 8: a. Root Nodes of the Bayes Net after the Second Step b. Locations of likely Hypotheses after the Second Step



```
Task Force/Catapault Battalion-B2739

Task Force/Catapault Battalion-B2739
   Hypotheses:
     HYP-T2718-Task Force:  0.91
HYP-T2722-Catapault Battalion:  0.47
     HYP-T2716-Task Force:  0.36
     HYP-T2708-Task Force:  0.36
HYP-T2724-Catapault Battalion:  0.13
```

Figure 9: a. Root Node of the Bayes Net after the Third Step b. Locations of likely Hypotheses after the Third Step



```
Brigade-B3230

        Brigade-B3230
Hypotheses:
    HYP-T3209-Brigade:  0.34
    HYP-T3211-Brigade:  0.34

Task Force/Catapault Battalion-B2739
   Hypotheses:
     HYP-T2718-Task Force:  0.93
HYP-T2722-Catapault Battalion:  0.55
     HYP-T2716-Task Force:  0.42
     HYP-T2708-Task Force:  0.33
HYP-T2724-Catapault Battalion:  0.11
```

Figure 10: a. Bayes Net after the Fourth Step

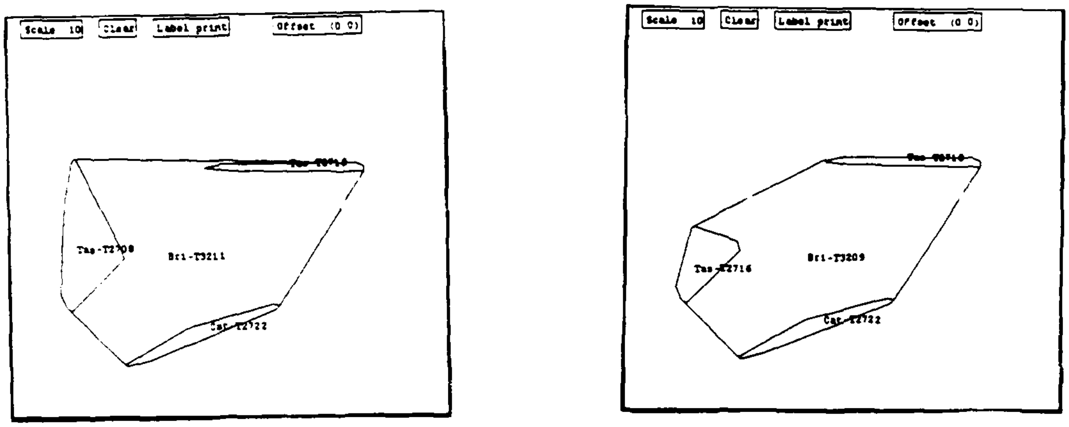
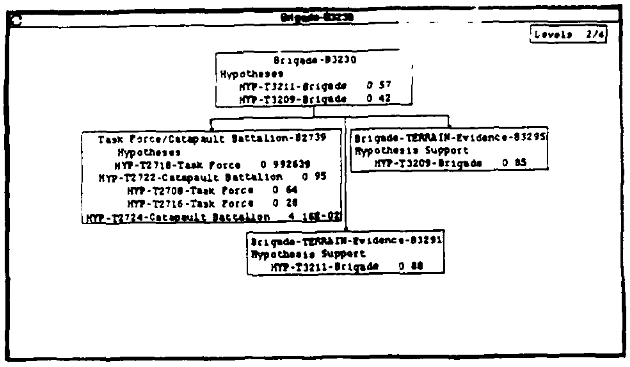


Figure 11: a. Location of First Brigade Hypothesis b. Location of Second Brigade Hypothesis



```
Brigade-B3230
Hypotheses
    HYP-T3211-Brigade    0 57
    HYP-T3209-Brigade    0 42

Task Force/Catapault Battalion-B2739
    Hypotheses
   HYP-T2718-Task Force    0 992639
 HYP-T2722-Catapault Battalion    0 95
      HYP-T2708-Task Force    0 64
      HYP-T2716-Task Force    0 28
HYP-T2724-Catapault Battalion   4 16E-02

Brigade-TERRAIN-Evidence-B3295
Hypothesis Support
    HYP-T3209-Brigade    0 85

Brigade-TERRAIN-Evidence-B3291
Hypothesis Support
    HYP-T3211-Brigade    0 88
```

Figure 12: Bayes Net after the Fifth Step

References

[Binford - 89] Binford, T.O., Levitt, T.S., and Mann, W.B., "Bayesian Inference in Model-Based Machine Vision", in Uncertainty in AI 3, Levitt, Kanal and Lemmer (Eds.), North Holland, 1989.

[Binford - 88] Binford, T.O., "Image Understanding: Intelligent Systems", *Proceedings of the DARPA Image Understanding Workshop*, 1988.

[Binford - 87] Binford, T.O., "Generic Surface Interpretation: Observability Model", *Proceedings of the International Symposium of Robotics Research*, 1987.

[Ettinger - 88] Ettinger, G.J., "Large Hierarchical Object Recognition Using Libraries of Parameterized Model Sub-parts", *Proceedings of the Computer Vision and Pattern Recognition Conference*, Ann Arbor, Michigan, June 1988.

[Franklin - 88] Franklin, J.E., Carmody, C.L., Keller, K., Levitt, T.S., Buteau, B.L., "Expert System Technology for the Military", *Proceedings of the IEEE*, 1988.

[Garey - 79] Garey, M.R., and Johnson, D.S., *Computers and Intractability: A Guide to the Theory of NP-Completeness*, W.H. Freeman and Company, San Francisco, California, 1979.

[Healey - 87] Healey, G., and Binford, T.O., "The Role and Use of Color in a General Vision System", *Proceedings of the International Conference on Artificial Intelligence*, August 1987.

[Healey - 88] Healey, G., and Binford, T.O., "A Color Metric for Computer Vision", *Proceedings of the DARPA Image Understanding Workshop*, 1988.

[Levitt - 88] Levitt, T.S., "Bayesian Inference for Radar Imagery Based Surveillance", *Uncertainty in Artificial Intelligence 2*, North-Holland, 1988.

[Nevatia - 73] Nevatia, R.K., and Binford, T.O., "Structured Descriptions of Complex Objects", *Third International Conference on Artificial Intelligence*, Stanford, California, 1973.

[Pearl - 86] Pearl, J., "Fusion, Propagation, and Structuring in Bayesian Networks", *Artificial Intelligence*, 1986.

[Ponce - 88] Ponce, J., and Healey, G., "Using Generic Geometric and Physical Models for Representing Solids", *Proceedings of the DARPA Image Understanding Workshop*, 1988.

[Shachter - 86] Shachter, R.D., "Evaluating Influence Diagrams", *Operations Research*, Vol. 34, No. 6, Nov-Dec 1986, pp. 871-882.

QUALITATIVE TARGET MOTION DETECTION AND TRACKING

Bir Bhanu, Peter Symosek, John Ming, Wilhelm Burger, Hatem Nasr, and Jon Kim

Honeywell Systems and Research Center
3660 Technology Drive
Minneapolis, MN 55418

ABSTRACT

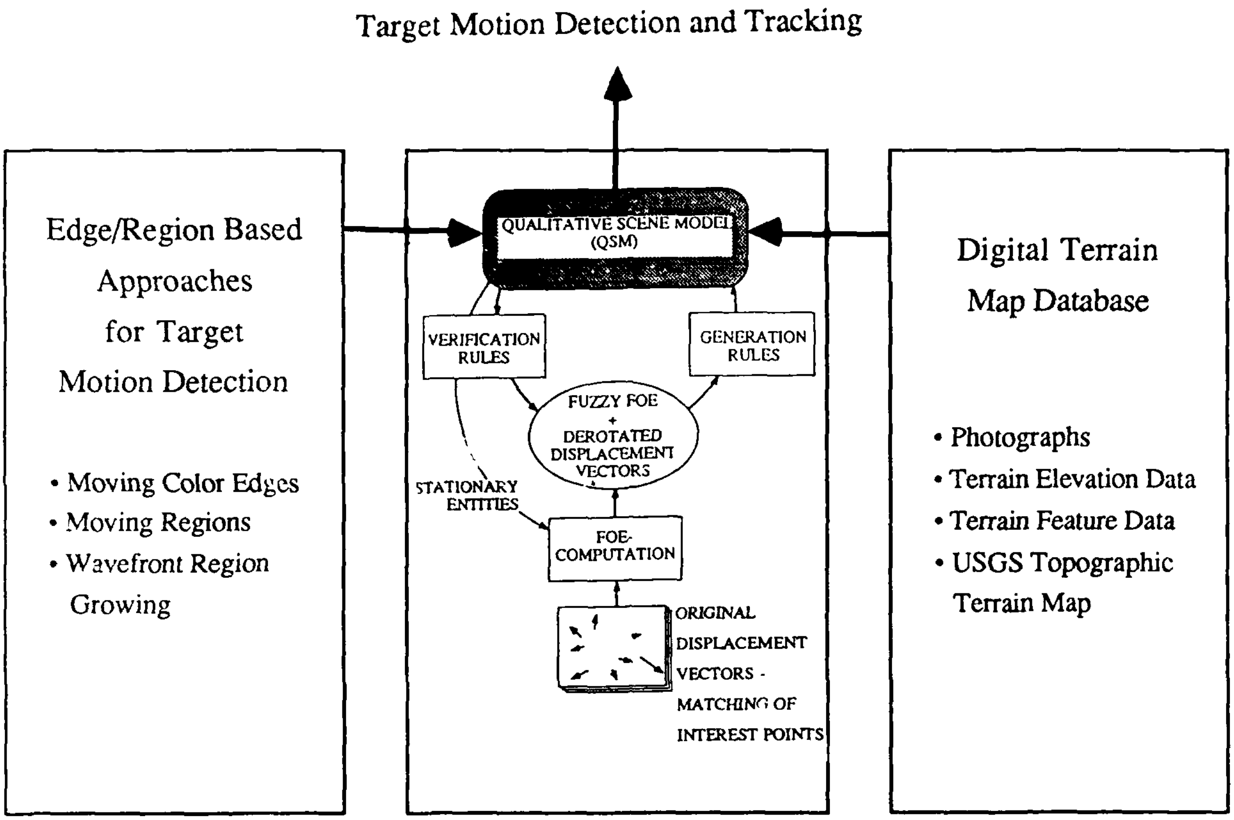
To detect and track moving targets using image information exclusively obtained from a vision system on-board an autonomous robotic vehicle, we present a comprehensive qualitative approach which allows for (a) determination of vehicle motion, (b) qualitative estimation of dynamic 3-D stationary structure, (c) detection and classification of the motion of individual objects in the scene using point, edge, and region features. The 3-D motion of targets is obtained from displacement vectors of point features without any knowledge about the underlying 3-D structure, discovering inconsistencies between the current state of the initial qualitative 3-D scene model and the changes actually observed in the scene, and by detecting moving edges and regions. We have also integrated map-based information into the system's reasoning framework. The digital map information, which consists of elevation, photographic, and terrain feature (roads, rivers, land cover, etc.), is used to predict target motion, track targets through occlusion, and assist in on/off road navigation by the robotic vehicle. The digital map information provides valuable clues when detecting moving targets in high clutter and low contrast environments. We present experimental results to demonstrate the capabilities of the system.

1. INTRODUCTION

Current multimode tracking approaches incorporate several possible techniques such as feature matching, correlation, centroid tracking, silhouette matching, and Kalman filtering for motion detection and tracking from a mobile platform.[2] These systems encounter problems in practical scenarios where conditions including arbitrary sensor platform motion, high clutter, low contrast, distant targets, sun angles (glare), variety of terrain features, vehicle maneuvers, countermeasures, target occlusion, and battlefield conditions pose significant threats to mission success. In some targeting scenarios, target location prediction is not enough for tracking; recognition of the target and clutter rejection algorithms may be required. At Honeywell, we are developing a comprehensive *qualitative* approach for target motion detection and tracking from a mobile platform.[3,4,6,8,10] Our objective is to achieve robust behavior in such a system. The key elements of our approach are shown in Figure 1.

The target motion detection and tracking problem can be viewed as the task of finding consistent and plausible 3-D interpretations for any change observed in the 2-D image sequence. Due to the motion of the Autonomous Land Vehicle (ALV), stationary objects in the scene generally do not appear stationary in the image, whereas moving objects are not necessarily seen in motion. The three main tasks in our approach for target motion detection and tracking are: (a) to estimate the vehicle's motion; (b) to derive the 3-D structure of the stationary environment; and (c) to detect and classify the motion of individual targets in the scene. These three tasks strongly depend on each other. The direction of travel (i.e. translation) and rotation of the vehicle are estimated with respect to stationary locations in the scene. The focus of expansion (FOE) is not determined as a particular image location, but as a region of possible FOE-locations called the *Fuzzy FOE*. We present a qualitative strategy of reasoning and modeling for the perception of 3-D space from motion information. Instead of refining a single quantitative description of the observed environment over time, multiple *qualitative interpretations* are maintained simultaneously.

The qualitative interpretations are built in three separate steps (see Figure 1). First, significant features (points, boundaries, corners, etc.) are extracted from the image and the 2-D displacement vectors are computed for this set of features. For the examples shown here, points were automatically selected and tracked between individual frames. The image database used to carry out the experiments is described in Section 2. The Interest Point detection and matching approach, which is a revised version of the Moravec interest point operator [18] and Barnard and Thompson's disparity analysis technique,[1] is optimized for low depression angle ALV imagery and is described in Section 3. During the second step, the vehicle's direction of translation, i.e. the Focus of Expansion (FOE), and the amount of rotation in space are determined. Almost all the necessary numerical computation is performed in the FOE computation stage, which is described in Section 4. The third step (2-D Change Analysis) constructs an internal 3-D model of the scene. Section 5 outlines the concepts and operation of this Qualitative Scene Model. Scene interpretations obtained from Qualitative Reasoning are validated with geometric reasoning and validation rules employing an auxiliary map database. The details of the map-based target tracking approach developed for the Scene Dynamics program are presented in Section 6. A preprocessing stage for the detection of rapidly moving objects in the field of view is incorporated.



Target Motion Detection and Tracking
Edge/Region Based Approaches for Target Motion Detection
• Moving Color Edges
• Moving Regions
• Wavefront Region Growing
QUALITATIVE SCENE MODEL (QSM)
VERIFICATION RULES
GENERATION RULES
FUZZY FOE + DEROTATED DISPLACEMENT VECTORS
STATIONARY ENTITIES
FOE-COMPUTATION
ORIGINAL DISPLACEMENT VECTORS - MATCHING OF INTEREST POINTS
Digital Terrain Map Database
• Photographs
• Terrain Elevation Data
• Terrain Feature Data
• USGS Topographic Terrain Map

Figure 1: Qualitative Reasoning and Modeling approach for target motion detection and tracking. From the original displacement vectors (obtained by matching corresponding point features), the Fuzzy FOE and the derotated displacement field are computed. The Qualitative Scene Model (QSM) is built in a hypothesize-and-test cycle by two sets of rules. Generation rules search for significant image events and place immediate conclusions (hypotheses) in the QSM. A set of environmental entities that are believed to be stationary is supplied by the QSM for use in the FOE computation. A digital map database interacts with the QSM to detect and track moving targets in the image. Edge/region based approaches are used detect rapidly moving objects at close range.

We have performed experiments on edge and region-based approaches for target motion detection in color imagery, to assist in obtaining robust performance from the Qualitative Reasoning system. The technical details of these algorithms are described in Section 7. Finally, in Section 8, the conclusions of this paper and our plans for future enhancements of the qualitative motion detection and tracking approach are presented.

2. DATABASE OF IMAGES

In order to verify the capabilities of the system shown in Figure 1, a large image database was generated and processed. In subsequent sections of this paper, we will illustrate some of the results. The processing consisted of five stages:

(1) Interest Point Detection,

(2) Disparity Analysis,

(3) Qualitative Reasoning for Motion Detection and Tracking,

(4) Hypothesis Verification using Auxiliary Map Information, and

(5) Edge/Region Motion Detection at Close Ranges.

The image database contains five sequences which represent configurations of the imaging system and target that often prove to be extremely challenging for traditional tracking approaches. The five configurations are:

(1) Motion detection for targets traveling directly towards and away from the ALV.

(2) Tracking targets through total occlusion.

(3) Tracking targets in high clutter.

(4) Tracking targets at long ranges.

(5) Tracking system verification with auxiliary map information.

The images were obtained from the ERIM/Martin Marietta Collage I database. Examples of each of the five configurations could be found in multiple sequences, but the objective was to resolve just one problematic scenario per sequence. Three examples of images from one of the five sequences are presented in Figure 2. The images were digitized at 0.5 second intervals. The images were digitized as gray scale because the demonstration of target tracking requires only *Interest Points* or points of significant change in multiple directions in the intensity function of the image and these locations can be detected using the luminance or Y component of the NTSC television signal. When moving edges or regions are employed for matching by the Disparity Analysis algorithm, color information is used as well.

The images were preprocessed with a 3x3 window average filter to attenuate digitization noise. The noise was due to a lack of sufficient bandwidth in the digitizer and resulted in 1 pixel duration pattern noise in the signal output. The 3x3 window average filter was sufficient for the attenuation of the noise so that the Interest Point detection algorithm performed reliably.

3. COMPUTATION OF DISPLACEMENT VECTORS

3.1 SELECTION OF INTEREST POINTS

The first stage of the motion algorithm suite is the detection of *Interest Points* or points where the image's intensity demonstrates a discontinuity in multiple directions of the eight neighbor directions of the quadruled pixel grid. Discontinuities of this kind are indicative of corners in 3-space. These locations are detected with a modified Moravec Interest Operator.[17,18] The operator derives "interestingness" employing the same approach that the Moravec Operator utilizes, but the revised operator is adaptive for expected range-dependent image features. The window dimension for the modified Interest Operator is established with

$$H = W = w_0 + 2\left\lfloor \frac{i * Int\_SF}{512} \right\rfloor , \qquad (1)$$

where

H = the height of the window,

W = the width of the window,

w_0 = the window dimension for the first row of the image,

i = the row coordinate of the current row of the image, 1 <= i <= 512,

Int_SF = a scale factor that controls how rapidly the detection operator window dimension varies with range,

$\lfloor x \rfloor$ = the greatest integer less than or equal to x.

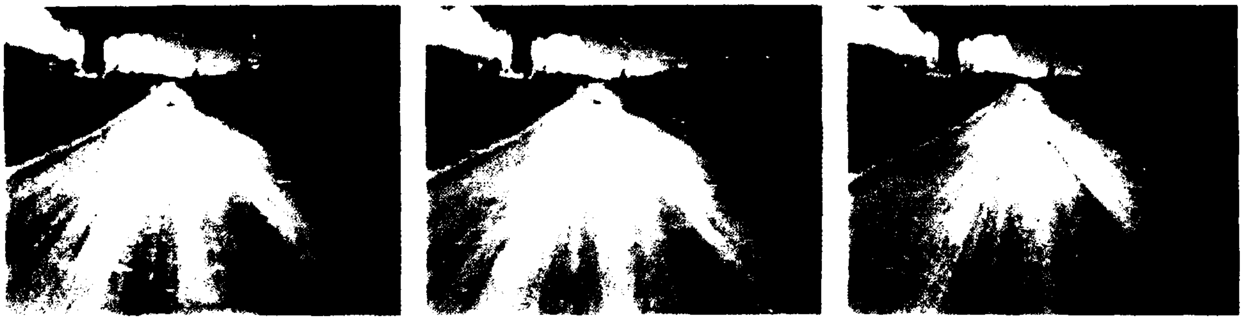


Figure 2: Three images of a sequence in which a car is approaching the ALV and another car is receding at a distance. This figure also shows the displacement vectors (by white lines) for Interest Points.

The approach, given by equation (1), essentially realizes a range adaptation capability by changing the window dimensions as a function of row coordinate. For low depression angle imagery, the range to 3-dimensional objects of the scene is approximately inversely proportional to the row coordinate of the image of the object. The resolution of scene regions that are seen in the lower portions of the image is greater than that of regions that are seen in the upper portions of the image because they are nearer to the ALV. Therefore, to detect features in these regions, the window dimension must increase to encompass enough image area for the entire neighborhood of the feature to be seen.

Because the scenes that comprise the NTSC video portion of the Collage I database are all on-road, the Moravec Interest Operator[18] detects fewer discontinuities in the higher resolution regions for several reasons. First, because of the rural nature of the Martin Marietta ALV Test Site, there are very few man-made objects by the side of the road (which normally exhibit good quality Interest Points), except for a few guard rails and telephone poles. Second, due to the scarcity of high quality interest points, the locations that are detected by the Moravec Interest Operator are often from regions displaying arbitrary textures such as small rocks or isolated stands of grass. The Moravec Interest Operator employs a cut off quota for point feature detection: The point features detected by the Interest Operator are stored in a list sorted in descending order of rating (a heuristic measure of the strength of the discontinuity). Only those interest points whose rating is greater than a cut off threshold are transferred to the next stage. Therefore, very few interest points for the lower regions of the image are archived in the locations of the list above the cut off quota.

In order to guarantee that the Interest Points the operator detects are not biased to any specific range, the revised operator does the following: The revised version of the algorithm uses a varying quota, where the quota is calculated as a function of the line coordinate of the image. This quota permits only $\frac{i}{512}$ * *Cut_Off* Interest Points to be detected for the first i lines of the image, where *Cut_off* is the cut off level of the original approach. When the number of Interest Points detected for the first i lines of the image is less than $\frac{i}{512}$ * *Cut_Off*, new Interest Points' quantified ratings are stored on the list in the usual fashion. When more than $\frac{i}{512}$ * *Cut_Off* Interest Points are detected for the first i lines of the image, these locations are integrated into the list only if their rating is greater than the ratings of the previously stored Interest Points. Otherwise, the new Interest Points are discarded.

3.2 DISPARITY ANALYSIS

The second stage of the Interest Point displacement estimation approach is disparity analysis. This algorithm is a revised version of the Barnard-Thompson Disparity Analysis algorithm.[1] The major difference between the Barnard-Thompson algorithm and the revised algorithm is enhanced feature matching that employs range adaptation. The phases of the algorithm where range adaptation is used are: 1) Neighbor Search, 2) Initial Match Likelihood Calculation, and 3) Relaxation. The revisions for each of these stages will be explained in the following sub-sections.

3.2.1 Neighbor Search

To restrain the size of the total set of candidate disparities and therefore, to diminish the computational complexity of the algorithm, the group of current frame Interest Points to which a previous frame Interest Point can be matched are those Interest Points lying in a W_S by H_S window centered at the previous frame location. The vertical and horizontal dimensions of this window change with the line coordinate of the previous frame Interest Point. The approach used to calculate the dimensions of this window is to estimate the maximum expected location change or disparity for an object at ground level for the imaging geometry of the ALV (as a function of line coordinate) and then empirically approximate the relation with an exponential function. The Neighbor Search window's dimension, obtained with this approach,[17] is:

$$W_S = H_S = 2\left\lfloor C_1 \exp[C_2 * i] + C_3 \right\rfloor + 1 \qquad (2)$$

where

W_S = the width dimension of the Neighbor Search window,

H_S = the height dimension of the Neighbor Search window,

$C_1 = 8.32 * S_{MAX}^{(-0.018868)}$,

$C_2 = \dfrac{\log\left[\dfrac{S_{MAX}}{8}\right]}{424}$,

$C_3 = w_0$, the Interest Operator for the first line of the image,

S_{MAX} = the maximum search window size for the near-field of the image (maximum expected object displacement),

i = the row coordinate of the Interest Point in the previous image.

3.2.2 Initial Match Likelihood Calculation

The window dimension for this stage is calculated by

$$W_M = H_M = 2\left[m_0 + \left\lfloor \frac{i * IM\_SF}{512} \right\rfloor\right] + 1 , \tag{3}$$

where

m_0 = Initial Match Likelihood window dimension for the first line of the image,
IM_SF = the Initial Match scale factor, and
i = the line coordinate of the current line.

The Initial Match Likelihood is calculated as a function of a heuristic measure of the correlation of the image intensities of the previous and current frames in windows centered at the location of the Interest Point in each frame. This heuristic measure will be explained in the latter paragraphs of this section. Because the relaxation algorithm used to refine the estimates of the disparity likelihoods employs probabilities (i.e. each disparity $l = [l_x, l_y]$ is characterized by a number $p(l)$, where $p(l)$ is an element of the range $[0,1]$ and $\sum_l p(l) = 1$), the Initial Match Likelihood is transformed into a probability using normalization. The heuristic measure is calculated for every candidate disparity l_j ; $j = 1,..., J_i$, where J_i is the total number of disparities identified in the neighbor search for a specific Interest Point i. Employing the notation of relaxation labeling schemes, each previous frame Interest Point represents a node or object in the object space of the iterative approximation procedure. Let a_i, $i = 1,...,Cut\_Off$ represent each node of the object space. The labels for each node a_i are $l_j, j = 1,..., J_i$ and an undefined disparity l^* (defined in the next paragraph). The set of all labels, $l_j, j = 1,..., J_i$, and l^* is denoted L_i. The objective of the iterative approximation procedure is to use evidence obtained from neighboring nodes of the object space to refine the estimated disparity likelihoods until one likelihood is approximately 1 and the rest 0, i.e. each object is uniquely labeled.

Because Interest Points are not detected with 100% certainty in each image, it is also possible that no valid match exists for each previous frame Interest Point. To account for this configuration, a non-match disparity l^* is defined. The likelihood $p_i(l^*)$ for node a_i represents the likelihood that the node i has no match. The first stage needed to transform heuristic correlation measures into probabilities is the calculation of the initial probability of the disparity $p_i(l^*)$. Using the approach of Barnard and Thompson,[1] the probability that an Interest Point is not matchable is approximated as $1 - w_i(l_{MAX})$, where $w_i(l_{MAX})$ is the heuristic measure of largest magnitude for node a_i. The assumption is justified due to the fact that the label of maximum weight is, in general, the correct one. We have verified this assumption for low-depression angle ground-to-ground imagery by empirically calculating the Initial Match Likelihood heuristic measure for a large number of frames and then tabulating the number of times that the true disparity was the one with the greatest measure.

The next stage is the application of Bayes Rule to obtain an initial estimate of the probability that a_i should be labeled l for labels other than l^*. This calculation is carried out as follows:

$$p_i^0(l_j) = p_i(l_j \mid i) * (1 - p_i^0(l^*)); \quad l_j \neq l^* . \tag{4}$$

where

$p_i^0(l_j)$ = the Initial Match Likelihood for disparity l_j of node a_i,
$p_i(l_j \mid i)$ = the conditional probability that a_i has label l_j given that a_i is matchable,
$(1 - p_i^0(l^*))$ = the probability that a_i is matchable.

The quantities $p_i(l_j \mid i)$ are estimated with

$$p_i(l_j \mid i) = \frac{w_i(l_j)}{\sum_{l_k \neq l^*} w_i(l_k)} . \tag{5}$$

In order to guarantee that the revised Disparity Analysis algorithm would perform correctly for a wide variety of scenarios, the following was done: A variety of match measures were evaluated for sequence 1 and the measure which produced the best results in terms of a qualitative visual evaluation of the estimated disparities was used to process the remaining four sequences. The qualitative visual evaluation was carried out by creating a pseudo-colored image (where one frame is displayed with the color green, the second is displayed with the color red and the estimated disparities are displayed with the color blue) and then visually checking the validity of the matches. The Initial Match measure which was judged to be the best, because it produced the fewest errors for a wide variety of scenarios, was

$$w_i(l_j) = \frac{1}{1 + C * S}\left[\frac{1 + L * I_k}{1 + L}\right]; \; i = 1,...,Cut\_Off; \; j = 1,..., J_i , \tag{6}$$

where

$w_i(l_j)$ = the initial match evidence of a candidate disparity l_j,

I_k = the k th Interest Point's rating, where the k'th Interest Point is the current frame location of the feature that defines disparity l_j, and $0 \leq I_k \leq 1$,

Cut_Off = number of Interest Points detected in the previous frame,

J_i = total number of potential disparities for Interest Point i,

C, L = constants, and

$$S = \frac{\sum_{j,k \,\varepsilon\, N} \left[g_{j_0+j,\, k_0+k} - h_{j_1+j,\, k_1+k} \right]^2}{\sum_{j,k \,\varepsilon\, N} 1} ,$$

where

$g_{m,n}$ = intensity of the previous frame at the location {m,n},

$h_{m,n}$ = intensity of the current frame at the location {m,n},

$\{j_0, k_0\}$ = previous frame coordinate of the point feature for disparity l_j,

$\{j_1, k_1\}$ = current frame coordinate of the point feature for disparity l_j, and

N = Initial Match region for disparity l_j.

3.2.3 Relaxation Region

With a relaxation labeling scheme, the valid disparities with low initial likelihoods are elevated in magnitude by the correlated evidence of their neighbors in the object space of the scene. By correlated evidence, it is meant that the disparities of neighboring nodes exhibit approximately the same magnitude and direction displacement. A node is a neighbor of another if it lies within the relaxation window centered at that node. Therefore, the correlated evidence for a specific disparity l_j of a_i is calculated as the sum of the disparity likelihoods for all neighbors of a_i, where the disparities of the neighbors have approximately the same magnitude and orientation as l_j. The degree of mismatch between the neighboring node's disparities and l_j is defined in terms of specific error thresholds for orientation and displacement. The Relaxation Region window's dimensions are calculated with this equation:

$$W_R = H_R = 2\left[r_0 + \left\lfloor \frac{i * R\_SF}{512} \right\rfloor \right] + 1 , \qquad (7)$$

where

W_R = the width dimension of the Relaxation Region window,

H_R = the height dimension of the Relaxation Region window,

r_0 = the Relaxation Region window dimension for the first line of the image,

R_SF = the scale factor for adaptation of the Relaxation Region window dimension with respect to range, and

i = the line coordinate of the current line of the image.

The iterative update rule for disparity probabilities, employed at each stage of the relaxation algorithm, is

$$\tilde{P}_i^k(l_j) = P_i^{k-1}(l_j) * (B + A * G^k(l_j)) \; ; \; j = 1,\ldots, J_i \qquad (8)$$

$$\tilde{P}_i^k(l^*) = P_i^{k-1}(l^*)$$

$$P_i^k(l_j) = \frac{\tilde{P}_i^k(l_j)}{\sum_{l \,\varepsilon\, L_i} \tilde{P}_i^k(l_m)}$$

where

$P_i^k(l_j)$ = the probability that a specific hypothesized disparity l_j, at the k'th iteration, is the true disparity,

$P_i^{k-1}(l_j)$ = the probability that a specific hypothesized disparity l_j, at the (k-1)st iteration, is the true disparity,

$G^k(l_j)$ – the correlated evidence obtained from other disparities lying in the Relaxation Region at iteration k,

A, B = constants that control the rate of convergence of the iterative procedure, and

R_i = the W_R x H_R Relaxation Region window centered at the previous coordinate of the point feature.

For the preceding equation, terms with tildes (˜'s) are evidences. Evidences are not constrained to be elements of the range [0,1] and therefore are not probabilities. Normalizing these terms by the total weight of all the evidences for a specific domain transforms evidences into probabilities.

3.3 RESULTS FOR DISPLACEMENT VECTORS

The adaptive window approach was validated with several sequences of real ALV imagery. Figure 2 shows the displacement vectors (white lines) for a few frames of a sequence in which the ALV was traveling along a paved road and, during the first half of the 8 second sequence, a vehicle passes the ALV on the left and, during the last half of the sequence another vehicle approaches the ALV in the opposite lane. Another example, given in Figure 3, demonstrates the estimated disparities for the points employed for matching for two successive frames of a sequence. The Interest Points were ranked on the basis of their interestingness, and the best 50 were selected, subject to the constraint that for each line of the image with coordinate i, no more than (i / 511) * 50 points were selected. By choosing a greater number of Interest Points for the lower half of the image than the upper half, the Interest Points detected represent valid object features and not arbitrary textures. The interest operator window size varied from 4x4 to 8x8. For a specific y coordinate, the interest operator window size is computed by

$$\text{Window Size}_X = \text{Window Size}_Y = 4 + \frac{4 * i}{512} \qquad (9)$$

The Neighbor Search window size varied from 4x4 to 120x120, the Initial Match Likelihood Calculation window size varied from 5x5 to 7x7, and the Relaxation Region window size varied from 32x32 to 128x128.

Another series of experiments was carried out with imagery obtained from a video camera on-board a low flying rotorcraft. The detection of obstacles such as low-hanging branches, trees, and power lines is critical for rotorcraft engaged in flying in Nap-Of-the-Earth (NOE) courses due to the substantial clearance required by their rotors. Figure 4 demonstrates the automated detection of Interest Points, the derivation of the valid disparities, and the calculation of the Fuzzy Focus of Expansion (FOE) (discussed in the next section). Each frame of the 246 frame database is processed with the revised Moravec Interest Operator and the revised Disparity Analysis algorithm for the estimation of disparities. The array of disparities for each pair of frames is transferred to the FOE computation stage.

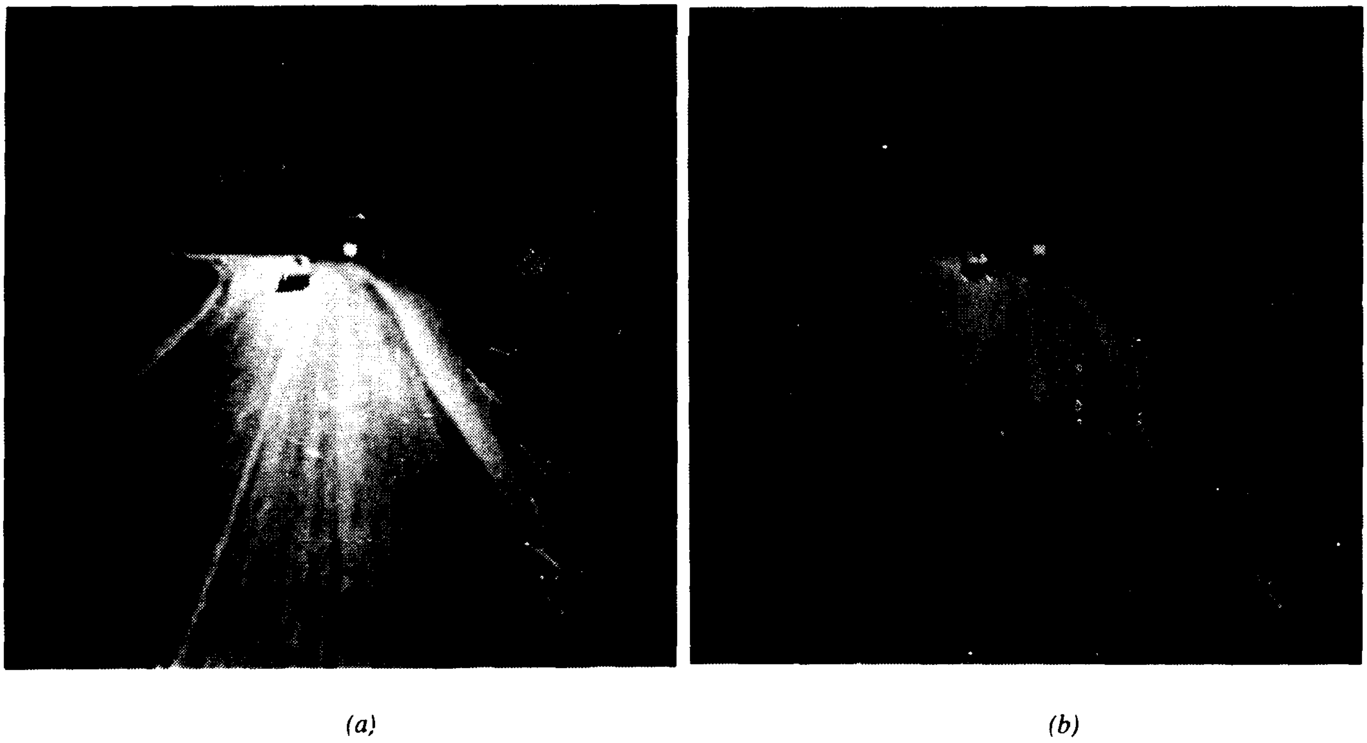
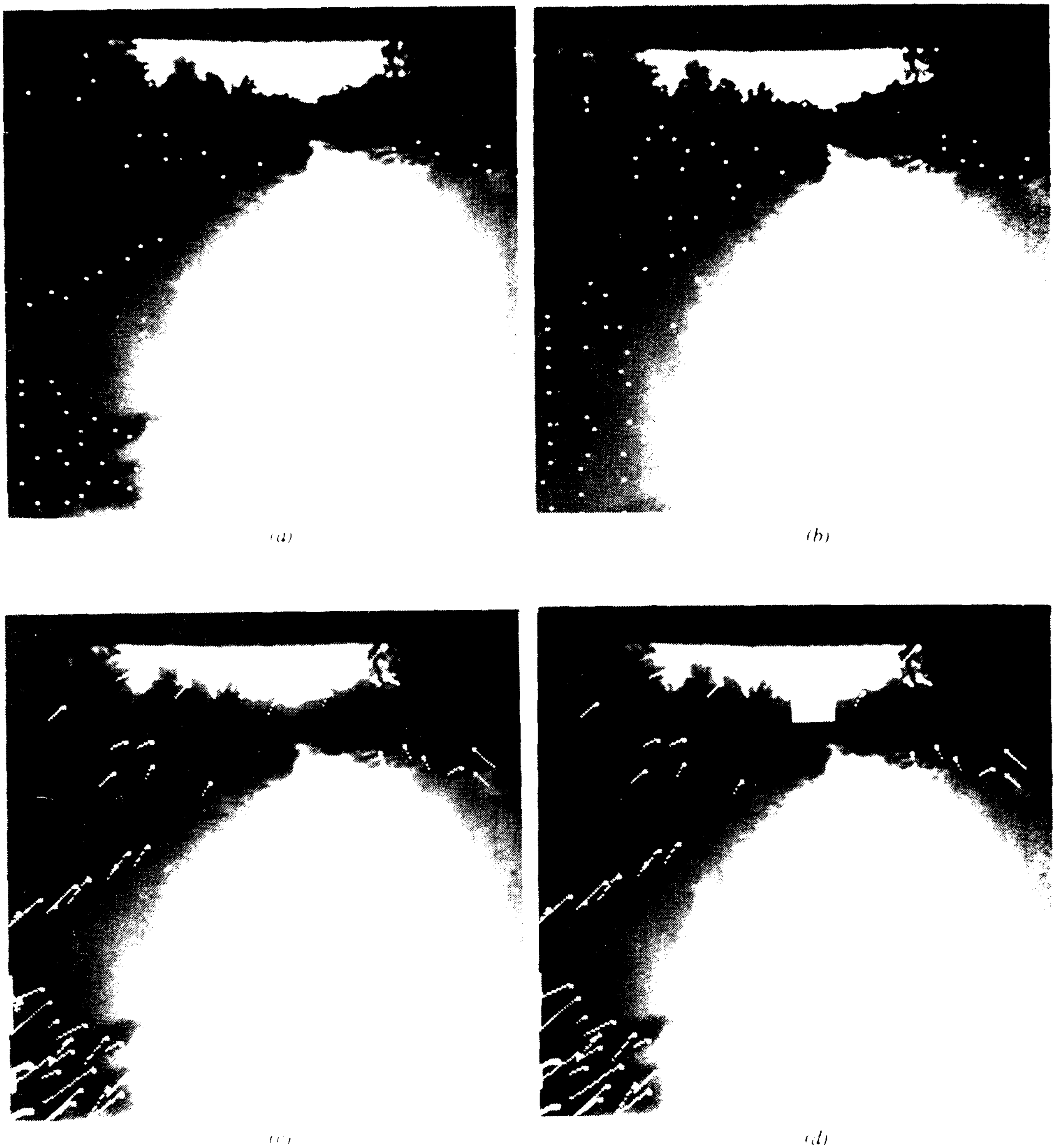

(a) (b)

Figure 3: Selection of Interest Points and disparity analysis. Displacement vectors are indicated by white lines. Note the displacement vectors present on the moving cars.



(a)

(b)

(c)

(d)

Figure 4: Disparity estimation for Nap-Of-the-Earth (NOE) rotorcraft data. *(a)* Interest Points in frame 1. *(b)* Interest Points in frame 2. *(c)* Displacement vectors. *(d)* Computation of Fuzzy Focus of Expansion. Fuzzy FOE is shown by the white area near the horizon. The black dot in the center indicates the best estimate of the FOE.

4. COMPUTATION OF THE FUZZY FOCUS OF EXPANSION

For short time intervals, the 3-D motion $\mathbf{M}$ of the ALV can be modeled by a translation $\mathbf{T}$ followed by a rotation $\mathbf{R}_\theta$ about the Y-axis and a rotation $\mathbf{R}_\phi$ about the X-axis:[9, 10]

$$\mathbf{M} = \mathbf{R}_\phi \, \mathbf{R}_\theta \, \mathbf{T} \ . \tag{10}$$

This results in a mapping $\mathbf{d}$ from the original image $\mathbf{I}_0$ at time t_0 into the new image $\mathbf{I}_1$ at time t_1.

$$\mathbf{d} : \mathbf{I}_0 \rightarrow \mathbf{I}_1 \ = \ \mathbf{r}_\phi \, \mathbf{r}_\theta \, \mathbf{t} \, \mathbf{I}_0 \ = \ \mathbf{r}_\phi \, \mathbf{r}_\theta \, \mathbf{I}_0' \ . \tag{11}$$

The intermediate image $\mathbf{I}_0'$ in (11) is the result of the translation component of the vehicle's motion and has the property of being a radial mapping, which deterministically is represented as:

$$\mathbf{t} = \{ \ (\mathbf{x}_i \ , \mathbf{x}_i') \in \mathbf{I}{\times}\mathbf{I}' \mid \mathbf{x}_i' = \mathbf{x}_i + \mu_i \, (\mathbf{x}_i - \mathbf{x}_f) \ , \mu_i \in R, \ \mu_i \geq 0 \ \} \ . \tag{12}$$

Unlike the two images $\mathbf{I}_0$ and $\mathbf{I}_1$, which are actually given, the image $\mathbf{I}_0'$ is generally not observed, except when the camera rotation is zero. It serves as an intermediate result to be reached during the separation of translational and rotational motion components. The fact that

$$\mathbf{I}_0' = \mathbf{r}_\theta^{-1} \, \mathbf{r}_\phi^{-1} \, \mathbf{I}_1 = \mathbf{t} \, \mathbf{I}_0 \tag{13}$$

suggests two different strategies for separating the motion components:

(1) **FOE from Rotation**: Successively apply combinations of inverse rotation mappings $\mathbf{r}_{\theta_1}^{-1} \, \mathbf{r}_{\phi_1}^{-1}$, $\mathbf{r}_{\theta_2}^{-1} \, \mathbf{r}_{\phi_2}^{-1}$,... $\mathbf{r}_{\theta_k}^{-1} \, \mathbf{r}_{\phi_k}^{-1}$ to the second image $\mathbf{I}_1$, until the resulting image $\mathbf{I}'$ is a radial mapping with respect to the original image $\mathbf{I}_0$. Then locate the FOE $\mathbf{x}_{f_k}$ in $\mathbf{I}_0$.

(2) **Rotation from FOE**: Successively select FOE-locations (different directions of vehicle translation) $\mathbf{X}_{f_1}$, $\mathbf{X}_{f_2}$,... $\mathbf{X}_{f_l}$ in the original image $\mathbf{I}_0$ and then determine the inverse rotation mapping $\bar{\mathbf{r}}_{\theta_l}^{-1} \, \bar{\mathbf{r}}_{\phi_l}^{-1}$ that yields a radial mapping with respect to the given FOE $\mathbf{x}_{f_l}$ in the original image $\mathbf{I}_0$.

Both alternatives were investigated under the assumption of restricted, but realistic vehicle motion. It turned out that the major problem in the **FOE-from-Rotation** approach is to determine if a mapping of image points is (or is close to being) radial when the location of the FOE is unknown. Of course, in the presence of noise, this problem becomes even more difficult. The second approach was examined after it appeared that any method which extends the given set of displacement vectors *backwards* to find the FOE is inherently sensitive to image degradations.

Although there have been a number of suggestions for FOE-algorithms in the past,[15, 20, 22] no results of implementations have been demonstrated on real outdoor imagery. One reason for the absence of useful results might be that most researchers have tried to locate the FOE in terms of a single, distinct image location. In practice, however, the noise generated by merely digitizing a perfect translation displacement field may keep the resulting vectors from passing through a single pixel. Even for human observers it seems to be difficult to determine the exact direction of heading (i.e., the location of the FOE on the retina). Average deviation of human judgement from the real direction has been reported[21] to be as large as 10° and up to 20° in the presence of large rotations.

It was, therefore, an important premise in this work that the final algorithm should determine an *area* of potential FOE-locations (called the *Fuzzy FOE*) instead of a single (but probably incorrect) point. The method described below avoids the problem mentioned above by guessing an FOE-location first and estimating the optimal derotation for this particular FOE in the second step.

4.1 FUZZY FOE ALGORITHM

Given the two images $\mathbf{I}_0$ and $\mathbf{I}_1$ of corresponding points, the main algorithmic steps of this approach are:[9]

(1) Guess an FOE-location $\mathbf{x}_f^{(i)}$ in image $\mathbf{I}_0$ (for the current iteration i).

(2) Determine the derotation mapping $\mathbf{r}_\theta^{-1}$, $\mathbf{r}_\phi^{-1}$ which would transform image $\mathbf{I}_1$ into an image $\mathbf{I}_1'$ such that the mapping $(\mathbf{x}_f^{(i)}, \mathbf{I}_0, \mathbf{I}_1')$ deviates from a radial mapping with minimum error $E^{(i)}$.

(3) Repeat steps *(1)* and *(2)* until an FOE-location $\mathbf{x}_f^{(k)}$ with the lowest minimum error $E^{(k)}$ is found.

An initial *guess* for the FOE-location is obtained from knowledge about the orientation of the camera with respect to the vehicle. For subsequent pairs of frames, the FOE-location computed from the previous pair can be used as a starting point. Once a particular $\mathbf{x}_f$ has been selected, the problem is to compute the rotation mappings $\mathbf{r}_\theta^{-1}$ and $\mathbf{r}_\phi^{-1}$ which, when applied to the image $\mathbf{I}_1$, will result in an optimal radial mapping with respect to $\mathbf{I}_0$ and $\mathbf{x}_f$.

To measure how close a given mapping is to a radial mapping, the perpendicular distances between points in the second image $(\mathbf{x}_i')$ and the "ideal" displacement vectors is measured. The sum of the squared perpendicular distances d_i

is the final error measure. For each set of corresponding image points ($\mathbf{x}_i \in \mathbf{I}$, $\mathbf{x}_i' \in \mathbf{I}'$), the error measure is defined as:

$$E\,(\mathbf{x}_f) = \sum_i E_i = \sum_i d_i^2 = \sum_i \left[\frac{1}{|\overrightarrow{\mathbf{x}_f\mathbf{x}_i}|}\, \overrightarrow{\mathbf{x}_f\mathbf{x}_i} \times \overrightarrow{\mathbf{x}_f\mathbf{x}_i'} \right]^2 . \tag{14}$$

The final algorithm for determining the direction of heading as well as horizontal and vertical camera rotations is the following:

(1) Guess an initial FOE $\mathbf{x}_f^0$, for example the FOE-location obtained from the previous pair of frames.

(2) Starting from $\mathbf{x}_f^0$, search for a location $\mathbf{x}_f^{opt}$ where $E_{\min}(\mathbf{x}_f^{opt})$ is a minimum. A technique of *steepest descent* is used, where the search proceeds in the direction of least error.

(3) Determine a region around $\mathbf{x}_f^{opt}$ in which the error is below some threshold.

The error function $E(\mathbf{x}_f)$ is computed in time proportional to the number of displacement vectors *N*. The final size of the FOE-area depends on the local shape of the error function and can be constrained not to exceed a certain maximum *M*. Therefore, the time complexity is *O(MN)*.

4.2 RESULTS FOR FOE COMPUTATION

Figure 4(d) shows the computation of Fuzzy FOE for the data taken from a rotorcraft during Nap-Of-the-Earth flight. Fuzzy FOE is shown by the white area near the horizon. The black dot in the center indicates the best estimate of the FOE.

5. QUALITATIVE REASONING AND MODELING FOR MOTION DETECTION AND TARGET TRACKING

5.1 QUALITATIVE REASONING TECHNICAL APPROACH

The choice of a suitable scheme for the internal representation of the scene is of great importance. The *Qualitative Scene Model* (QSM) is a 3-D camera-centered interpretation of the scene that is built incrementally from visual information gathered over time. The nature of this model, however, is *qualitative* rather than a precise geometric description of the scene. The basic building blocks of the QSM are *entities*, which are the 3-D counterparts of the 2-D *features* observed in the image. For example, the point feature *A* located in the image at *x,y* at time *t*

(Point_Feature *A t x y*)

has its 3-D counterpart in the model as

(Point_Entity *A*).

Since the model is camera-centered ("retinocentric"), the image locations and 2-D movements of features are implicitly part (i.e., known facts) of the model. Additional entries are the properties of entities (e.g. "stationary" or "mobile") and relationships between entities (e.g. "closer"), which are not given facts but hypotheses about the real scene. This is expressed in the model as either

(Stationary *entity*) or (Mobile *entity*) .

It is one of the key features of the QSM that it generally contains not only one interpretation of the scene, but a (possibly empty) *set* of interpretations which are all pursued simultaneously. At any point in time, a hypothesis is said to be "feasible" if it exists in the QSM and is not in conflict with some observation made since it was established.

Interpretations are structured as an inheritance network of partial hypotheses. Individual scene interpretations are treated as "closed worlds", i.e., a new conclusion only holds within an interpretation where all the required premises are true. Interpretations are also checked for internal consistency, e.g. entities cannot be both stationary *and* mobile within the same interpretation. The QSM is maintained through a generate-and-test process as the core of a rule-based blackboard system. The two major groups of rules are: *Generation Rules* and *Verification Rules.*

Generation Rules

Generation rules examine the (derotated) image sequence for significant changes and modify each interpretation in the QSM. Some of these observations have unconditional effects upon the model. For example, if an image feature is found to be moving *towards* the Fuzzy FOE (instead of diverging away from it), then it belongs to a moving entity in 3-D space. The actual rule contains only one premise and asserts (MOBILE ?x) as a global fact (i.e., it is true in every interpretation):

```
(defrule DEFINITE_MOTION
  (MOVING_TOWARDS_FOE ?x ?t)
  =>
  (at ROOT (assert (MOBILE ?x)))) /*a global fact*/
```

The directive "at ROOT" places the new fact at the root of the interpretation graph, i.e., it is inherited by all existing interpretations.

Other observations depend upon the facts that are currently true in a "world" and, therefore, may have only local consequences inside particular interpretations. For example, if two image features *A* and *B* lie on opposite sides of the Fuzzy FOE and they are getting closer to each other, then they must be in relative motion. If an interpretation exists that considers at least one of the two entities (x,y) stationary, then (at least) the other entity cannot be stationary (i.e., it must be mobile). The following rule "fires within" each interpretation that considers the first entity (x) stationary:

```
(defrule RELATIVE_MOTION
  (OPPOSITE_FOE ?x ?y ?t) /* first observation */
  (CONVERGING ?x ?y ?t)   /* second observation */
  (STATIONARY ?x) /* true inside an interpretation */
  =>
  (assert (MOBILE ?y)))   /* local to this interpretation */
```

Verification Rules

While the purpose of the generation rules is to establish new hypotheses and conclusions, the purpose of *verification rules* is to review interpretations after they have been created and, if possible, prove that they are false. When a hypothesis is found to be inconsistent with some new observation, it is usually removed from the QSM. Any interpretation that is based on such a hypothesis is removed simultaneously. Since we are always trying to come up with a single (and hopefully correct) scene interpretation, this mechanism is important for pruning the search tree.

Verification rules are typically based on image observations that, used as generators, would produce a large number of unnecessary conclusions. For example, the general layout of the scene seen from the top of a land-based vehicle suggest the rule of thumb that things which are *lower* in the image are generally closer to the camera. Although this rule is not strong enough to draw direct conclusions, it may be used to verify existing hypotheses:

```
(defrule LOWER_IS_CLOSER_HEURISTIC
  (CLOSER ?x ?y)
  (BELOW_THE_HORIZON ?x ?t)
  (BELOW_THE_HORIZON ?y ?t)
  (BELOW ?y ?x  ?t)
  =>
  /*mark this interpretation as conflicting*/
  (assert (CONFLICT LOWER/CLOSER ?x ?y))).
```

Whenever an existing hypothesis (CLOSER ?x ?y) violates the above rule of thumb, this rule fires and marks the interpretation as conflicting. How the conflict is eventually resolved depends upon the global state of the QSM. Simply removing the afflicted interpretation would create an empty model if this interpretation was the only one. This task is handled by a set of dedicated *conflict resolution rules.*[3]

The kind of rules described up to this point are mainly based upon the geometry of the imaging process, i.e., perspective projection. Other important visual clues are available from occlusion analysis, perceptual grouping, and semantic interpretation. *Occlusion* becomes an interesting phenomenon when features of higher dimensionality than points are employed, such as lines and regions. Similarities in form and motion found by *perceptual grouping* allow us to assemble simple features into complex objects. Finally, as an outcome of the recognition process, *semantic* information may help to disambiguate the scene interpretation. If an object has been recognized as a building, for example, it makes every interpretation obsolete that considers this object mobile. For all these various lines of reasoning, the QSM serves as a common platform.

Meta Rules

In summary, the construction of the QSM and the search for the most plausible scene interpretation are guided by the following meta rules:

- Always tend towards the "most stationary" (i.e. most conservative) solution. By default all new entities are considered stationary.
- Assume that an interpretation is feasible unless it can be proved to be false (the principle of "lack of conflict").
- If a new conclusion causes a conflict in one but not in another current interpretation, then remove the conflicting interpretation.
- If a new conclusion cannot be accommodated by any current interpretation, then create a new, feasible interpretation and remove the conflicting ones.

More details about QSM and rules are given in the Dynamic Reasoning using Integrated Visual Evidence (DRIVE) technical report.[3]

5.2 QUALITATIVE REASONING RESULTS

The computation of the Fuzzy Focus of Expansion and Qualitative Scene Model are implemented on a Symbolics 3670. Qualitative Reasoning is implemented in a knowledge-based system development environment, called the Automated Reasoning Tool (ART), which is supplied by Automated Inference Corp.

Figure 5 presents four frames from the Collage I database, where a vehicle is seen traveling down on the very distant roadway in the top right corner of the image. The distance from the ALV to the other vehicle is several hundred feet. The four frames were processed and the estimated Interest Point disparities between the first and second frames, the second and third frames and the third and fourth frames are shown in Figures 6(a)-(c). Note that the target is detected in the images even though it is quite far away from the ALV.

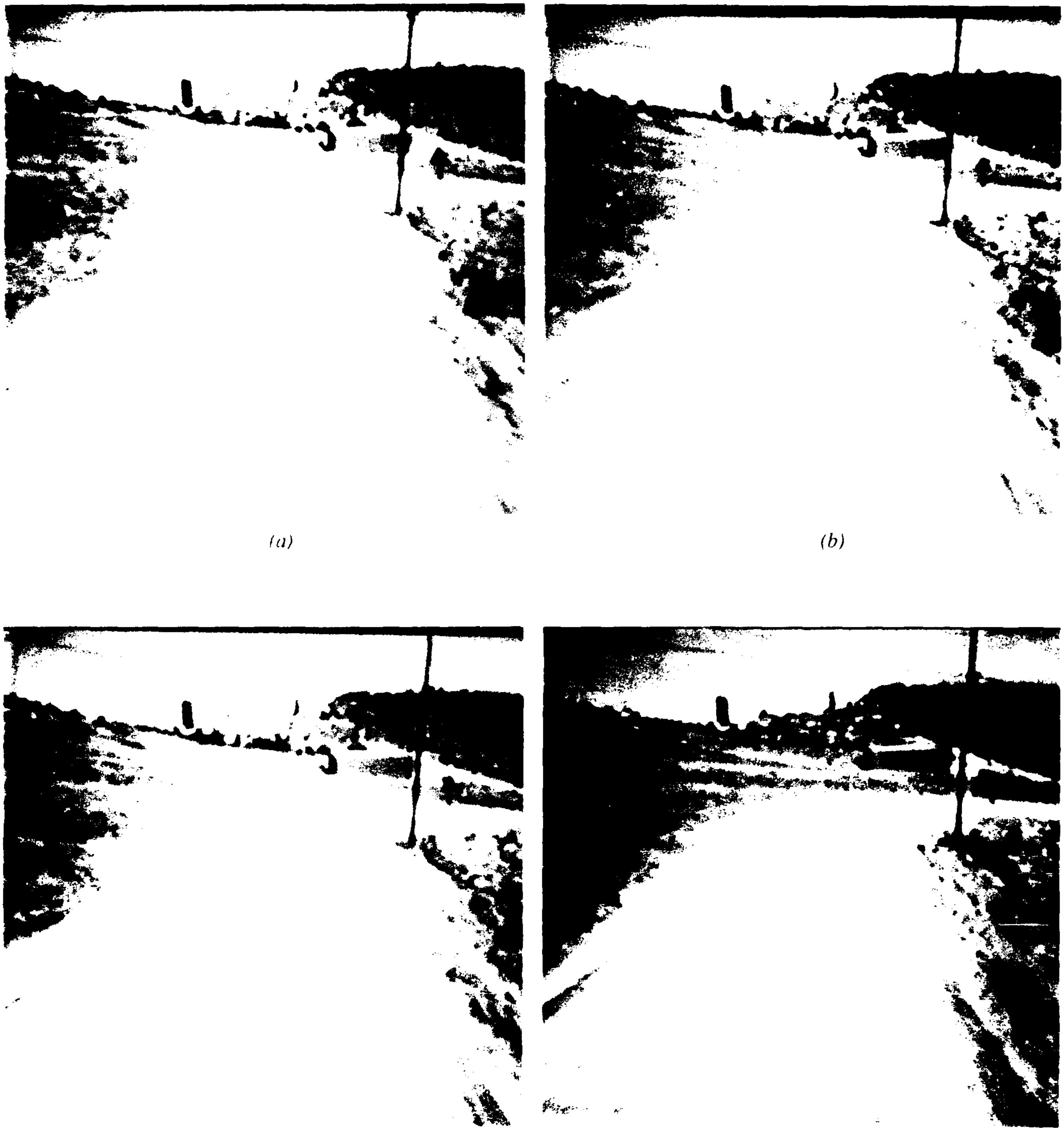
The estimated disparities for each pair of images are then transferred to the FOE location estimation stage. The results obtained for the first pair of images (frames 15 and 16) in Figure 5 are shown in Figure 7, where Figure 7(a) depicts the Interest Point locations in the coordinate frame of the second image of the pair along with the estimated disparity of the point (shown as a line attached to the apex of the pointer to the Interest Point location) and Figure 7(b) is the result of Qualitative Scene Model Calculation. The results obtained for the second and third and third and fourth frames of this example are presented in Figures 8 and 9, respectively.

6. MAP-BASED TARGET TRACKING

Target motion detection and tracking is essential for the potential military applications of a robotic vehicle. However, purely image-based target motion information may have restricted use in many practical military scenarios such as reconnaissance, where map location (latitude, longitude, and elevation) of the moving targets, precise information on their direction of movement, and knowledge about nearby roads and terrain may be crucial. We have developed and implemented the Map Assisted Tracking System (MATS), which integrates the digital terrain map information with Honeywell's Qualitative Reasoning system,[5,7] which was described in the last three sections. At present, MATS is loosely integrated with the Qualitative Reasoning system to provide a comprehensive set of information about the map location of the moving objects, the road label that the targets are possibly traveling on, and neighboring landmarks. Beyond practical mission considerations, digital terrain map information can be very helpful in detecting and tracking moving targets in high clutter and low contrast scenes.

Figure 10 provides a high level view of the MATS system. As shown at the top of the figure, Qualitative Reasoning provides the motion direction, x and y image location, and relative range of the moving targets. MATS performs an image-to-map correspondence for the individual targets and determines the approximate location of the targets in the map. An uncertainty area is computed for each target. Then, the digital roads file is searched to locate nearby roads. A search algorithm determines the most likely roads and nearby landmarks. This computation requires that the vehicle's position in the map must be given, which can be obtained from the Land Navigation System (LNS) or an Inertial Navigation System (INS), for future robotic vehicles. MATS assumes a given camera view angle, which can be obtained from the gimbal controller. Road and terrain information is used to predict obscuration so that the Qualitative Reasoning system is able to track targets in high clutter scenarios.

The prototype of the MATS system has been implemented and initial tests have been performed in conjunction with the Qualitative Reasoning system. Currently, MATS is functional, although it needs to be fully integrated with the Qualitative Reasoning system in an end-to-end experiment. The image-to-map correspondence algorithm must be refined and further testing needs to be done on the entire system. The remainder of this section describes the characteristics of the digital map data that we have used and explains the details of its use in the MATS system and experimental results for two representative scenarios.



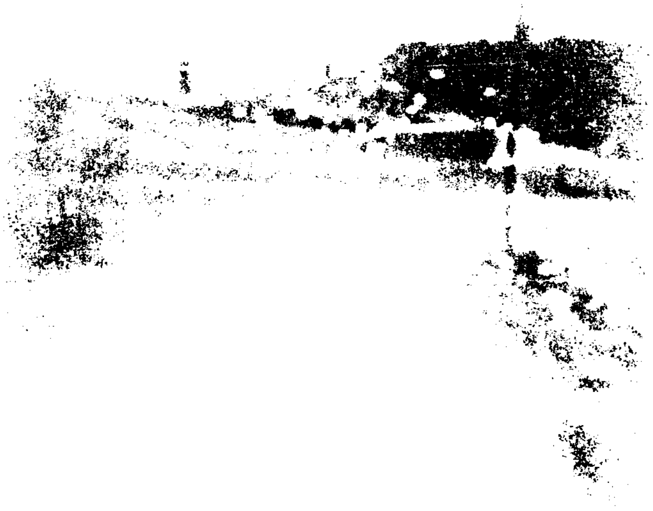
(a)

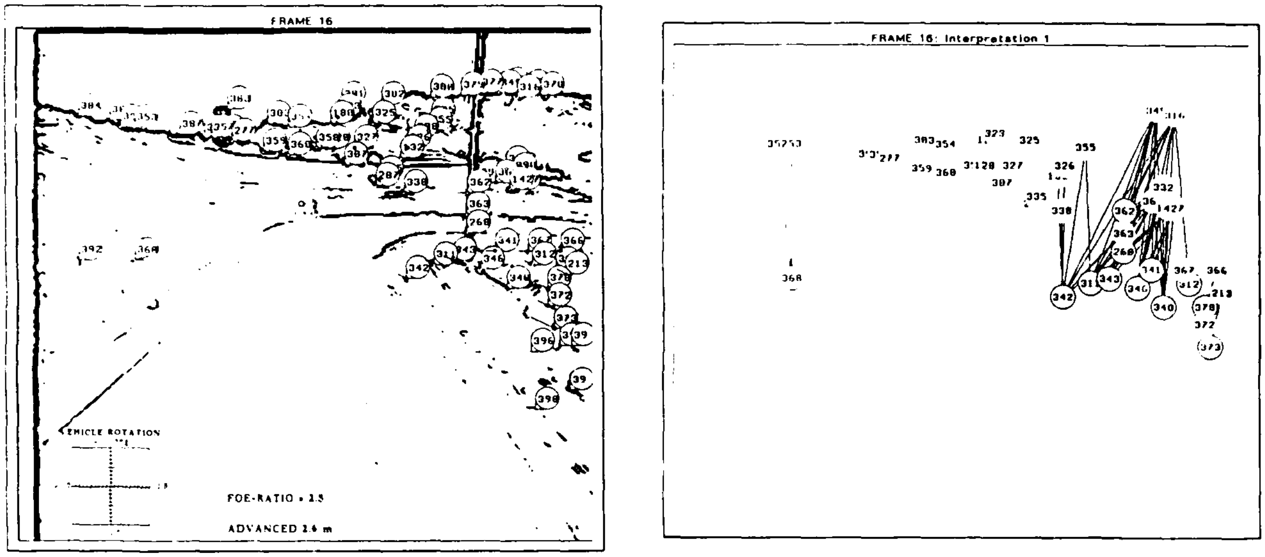
(b)

(c)

(d)

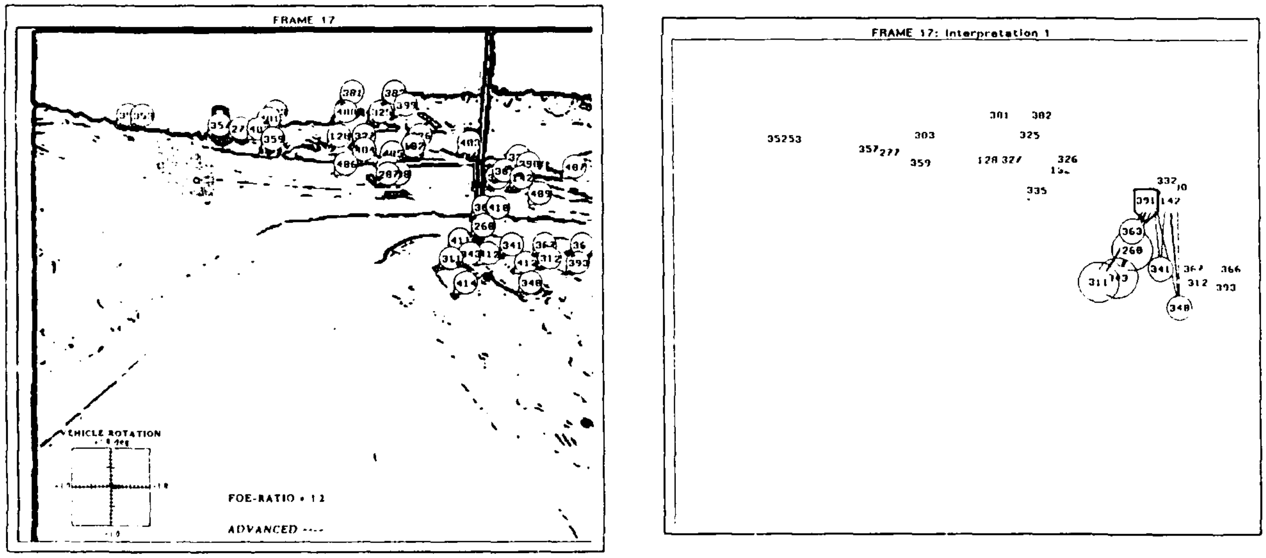
Figure 5: Four frames (15, 16, 17, 18) from the Collage I database, where a vehicle is seen traveling down in the very distant roadway in the top right corner of the image. *(a)* Frame 15. *(b)* Frame 16. *(c)* Frame 17. *(d)* Frame 18.





(a) (b)

Figure 7: Fuzzy FOE Computation and Qualitative Scene Model. *(a)* The Fuzzy FOE is shown by the shaded area. The circle within the shaded area is the most probable location for the FOE. *(b)* Qualitative Scene Model, where the size of one circle denotes its distance from the vehicle and links between circle or points indicate that closer relationships have been established between stationary entities.



(a) (b)

Figure 8: Fuzzy FOE Computation and Qualitative Scene Model. *(a)* The Fuzzy FOE is shown by the shaded area. The circle within the shaded area is the most probable location for the FOE. *(b)* Qualitative Scene Model, where the size of one circle denotes its distance from the vehicle and links between circle or points indicate that closer relationships have been established between stationary entities. Note that object 391 has been detected as moving. The direction of the arrow indicates that the target is moving downward.

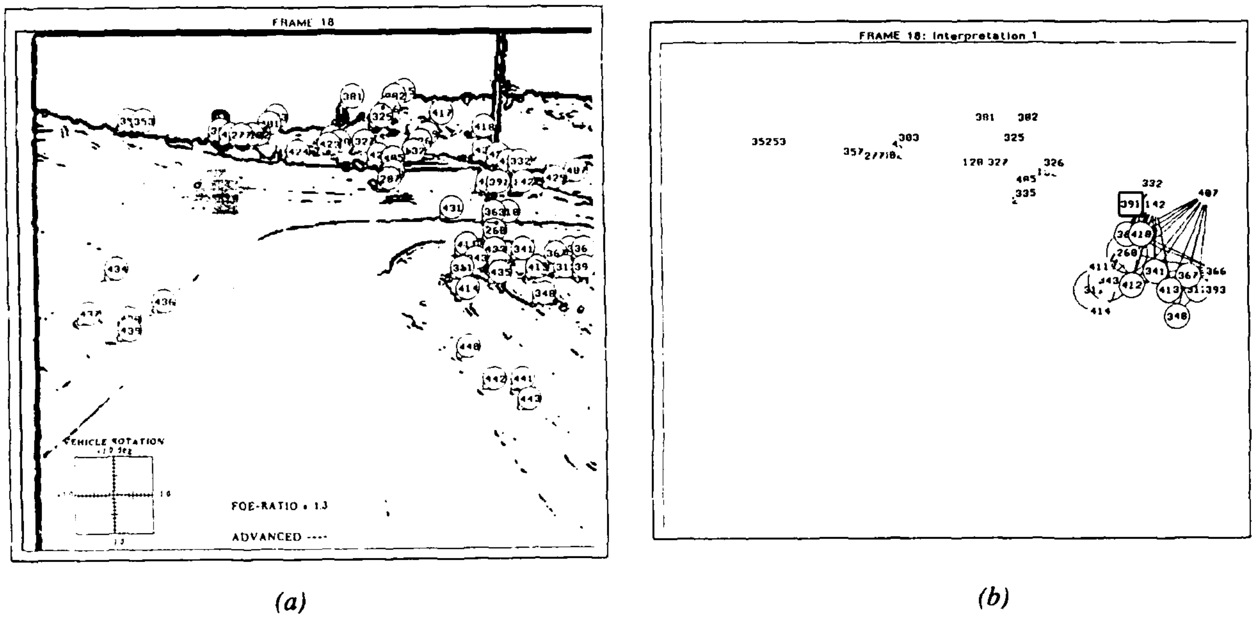


Figure 9: Fuzzy FOE Computation and Qualitative Scene Model. *(a)* The Fuzzy FOE is shown by the shaded area. The circle within the shaded area is the most probable location for the FOE. *(b)* Qualitative Scene Model, where the size of one circle denotes its distance from the vehicle and links between circle or points indicate that closer relationships have been established between stationary entities. Note that the target has been detected as moving, but its direction of movement could not be determined.

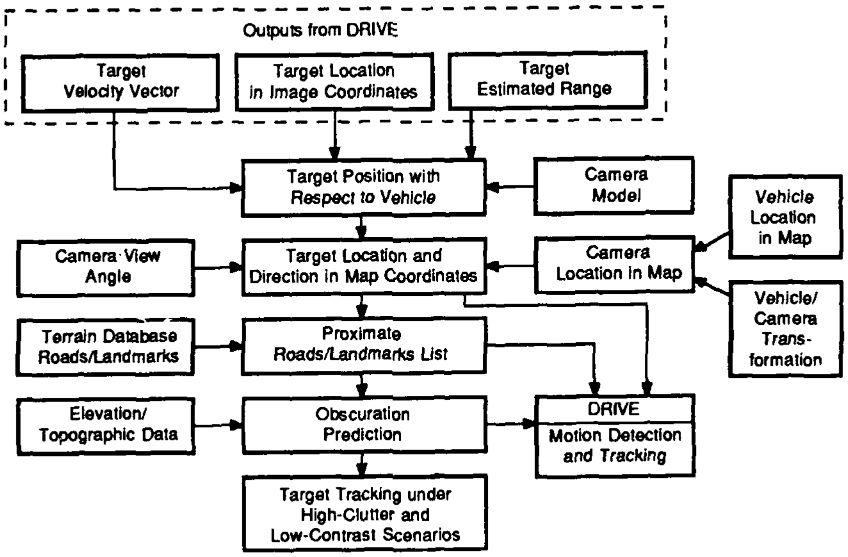

Outputs from DRIVE
Target Velocity Vector
Target Location in Image Coordinates
Target Estimated Range
Target Position with Respect to Vehicle
Camera Model
Vehicle Location in Map
Camera View Angle
Target Location and Direction in Map Coordinates
Camera Location in Map
Vehicle/ Camera Trans- formation
Terrain Database Roads/Landmarks
Proximate Roads/Landmarks List
Elevation/ Topographic Data
Obscuration Prediction
DRIVE
Motion Detection and Tracking
Target Tracking under High-Clutter and Low-Contrast Scenarios

Figure 10: Map Assisted Tracking System (MATS) which has been integrated with the DRIVE system for motion detection and tracking.

6.1 DIGITAL TERRAIN MAP DATABASE

The existing digital terrain map database of the Autonomous Land Vehicle (ALV) test site contains elevation, photographic, and terrain feature data. The elevation data specifies the elevation in meters above sea level at a given map position. The horizontal resolution of the elevation data is five meters per pixel. The photographic data includes a digitized aerial photograph of the ALV test site. The terrain data includes road, river, land cover, and soil information for the area. Only the road feature data from the terrain database was used in our MATS experiments. The road data consists of unique identifiers for each road segment. The roads are represented by their width and segment endpoint coordinates.

Each elevation data point in the original database was represented as 16-bit data. Since the actual variation in elevation over the mapped region was only 901 feet, we derived an 8 bit per pixel representation of the data that had units of meters. After the elevation data was derived, we transformed the feature data into the correct scale.

The terrain feature data files are provided as character files. Each line segment lists a brief header which includes a brief description, a unique identifier, and the number of line segments that belong to that segment, followed by the segment end points. The first step was to scale the endpoints to the elevation data and to run a line tracing algorithm to convert the data into vector form.

Using the digitized photograph at 4096 x 4096 resolution, a USGS 7.5 minute topographical terrain map, and the elevation data, prominent landmarks were identified and the correspondence was manually established. The road features were primarily used to establish the correspondence. Registration was done manually by selecting four corresponding points in each file. The four points were selected at the outer boundaries. Although the digitized aerial photographs were not orthophotos, they were treated as such for the purpose of these initial experiments. Future experiments would be better served by the usage of sensor and platform data to eliminate the effects of relief distortion and the resulting misalignment between the map and the image. Since the photograph data had a much finer resolution than the elevation data, the elevation data was bi-linearly interpolated to match the resolution of the photographic data. This resulted in a 0.7 meter spacing between posts in the elevation data. The same interpolation was performed on the road data. However, there was a trade-off to be made here between processing time and achievable resolution. It was decided that very fine resolution was not critical to the experiment and the data was sampled at a 1.4 meter resolution between posts.

Figures 11-13 show the digital terrain database which includes grid elevation data, elevation intensity data, and terrain features overlays.

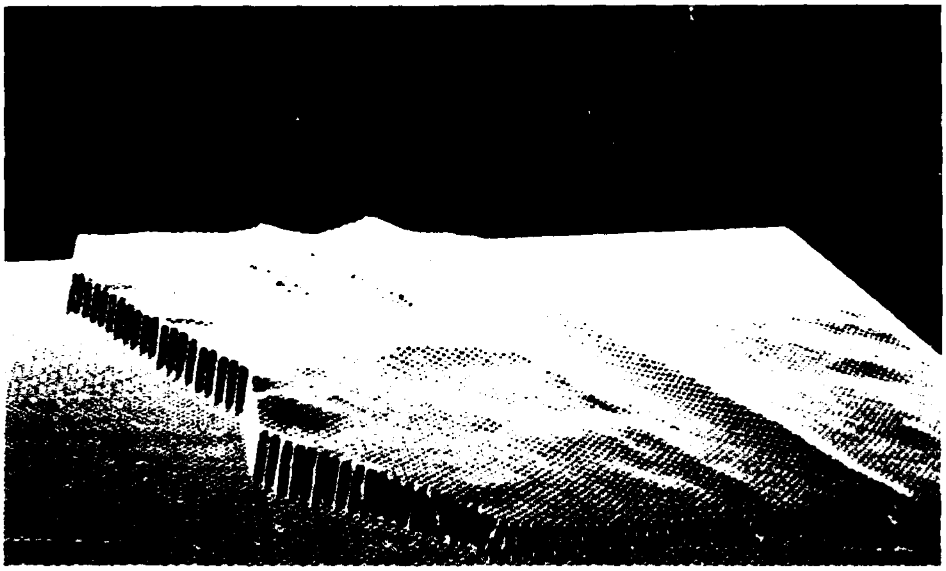


Figure 11: Digital elevation map of the ALV test site area displayed in a grid format.

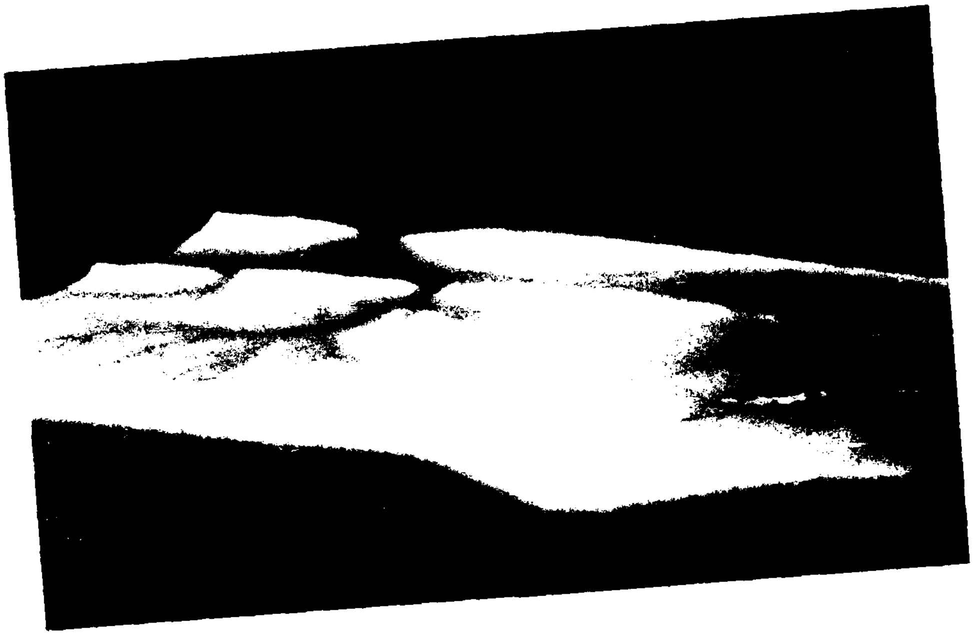


Figure 12: Digital Elevation Model (DEM) with overlays showing roads (black) and rivers.

Figure 13: Close-up of digital elevation map with all feature data overlayed.

6.2 MAP ASSISTED TRACKING SYSTEM DESCRIPTION

The MATS functional objectives are: 1) locate the 3-D map coordinates of moving targets, 2) identify the roads or terrain that they are traveling on, and 3) identify nearest landmarks to the targets. MATS assumes that the following information is given:

- Target (x,y) image location and traveling direction in image coordinates
- Range to target
- Camera model and a fixed view angle
- Digital terrain map database

The first task that MATS performs is to find a view vector from the vehicle's camera center to the target. As shown in Figure 14, the target is detected at T(x,y) location and the center of the image C(256,242) is where the view vector coincides with the optical axis of the camera. The equation of an arbitrary view vector can be determined given the imaging system's 3-D coordinates (this is obtained from the Inertial Navigation System data available on a full-scale robotic vehicle) and orientation.

For the current implementation of MATS, the estimates for the ALV's Universal Transverse Mercatur (UTM) coordinates are obtained as follows: The image coordinates of three landmarks, observed in the image, are noted and the view vectors to the landmarks in a vehicle-centered coordinate frame are calculated. Because the absolute location of each of the landmarks in terms of UTM coordinates is known, the distance between each pair of landmarks is known and the subtended angle between any pair of neighboring legs of the triangle defined by the landmarks is known. Because the geometry of this triangle is known, it is possible to estimate the distance from the imaging system to each of the landmarks. The coordinate of the ALV in the UTM coordinate space is obtained by solving for the unique 3-D location that is the prerequisite distance from each of the three landmarks. This also establishes the orientation of the ALV in UTM coordinates. Improved accuracy is obtainable by using 4-10 landmarks for calculation of the ALV's coordinates (employing least squares techniques).

When the orientation and location of the ALV is determined in terms of the UTM coordinate system, the location of the target is estimated with the following approach. Let the orientation of a view vector in terms of the camera coordinate system be expressed by the pure horizontal rotation $\mathbf{R}_\theta$ and pure vertical rotation $\mathbf{R}_\phi$ as shown in Figure 15. The θ and ϕ required to bring the target location to align with the optical axis are obtainable by estimating where the line of sight ray (defined by the view vector) intersects the digital terrain elevation map.

Let $(\Delta x, \Delta y, \Delta z)$ *be the displacement vector from the target to the vehicle*, then the target's real location in the map coordinates, (X_w, Y_w, Z_w), is:

$$X_w = X_V + \Delta x \,, \quad Y_w = Y_V + \Delta y \,, \quad Z_w = Z_V + \Delta z \tag{15}$$

where (X_V, Y_V, Z_V) is the coordinate of the current vehicle location in the map.

There are obviously uncertainties about the current vehicle map location and the location of the detected moving targets. In addition, there are uncertainties about the estimated range to the target obtained from Qualitative Reasoning. The terrain map information is used to correct such uncertainties. The initial hypothesis that targets are moving on roads allows MATS to search for the roads nearest to the computed target map location. The road file is represented as

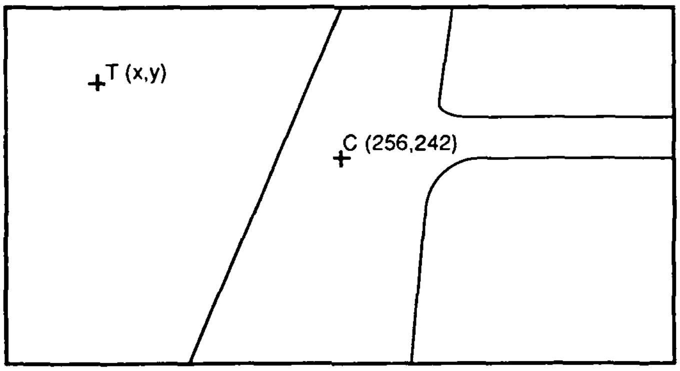


Figure 14: Orientation for locating target view vectors.

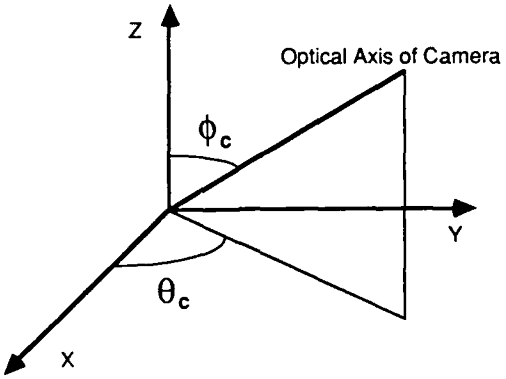


Figure 15: Orientation of the camera system.

a 2-D image file, where pixel values represent roads labels. Pixel values of zero indicate no roads, otherwise pixels values correspond to roads labels. From this road map representation, another file is generated which contains the shortest distances from each pixel to a road, and this file is pre-compiled to allow very rapid search. Once the target location is estimated in the map, MATS quickly searches for the x and y coordinates values in the road map representation and infers the road that the target is traveling on.

6.3 EXPERIMENTAL RESULTS

Experiments with MATS have been conducted on two scenarios (Figures 5 and 16). In both experiments, the targets detected were moving at a relatively far distance from the vehicle. Typically, the ranges were several hundred feet. Both scenarios contained high clutter and the contrast of the targets was low. In both experiments, only one target was moving in the image. Figures 17 and 18 show results calculated by MATS for the first example scenario and Figures 19 and 20 display the results for the second experiment.

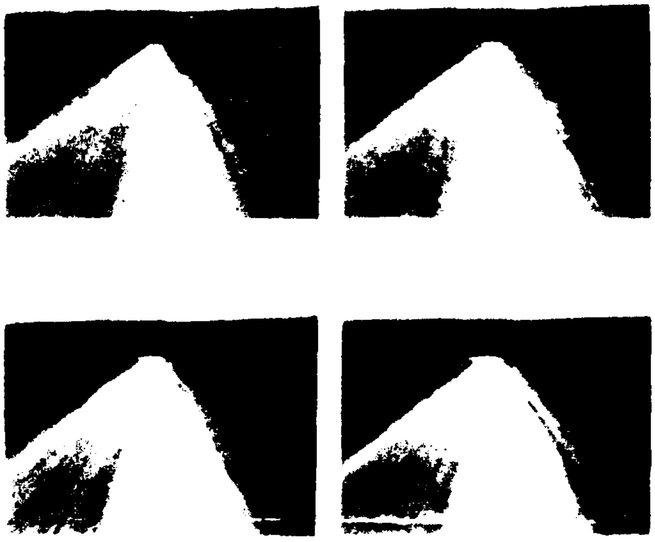


Figure 16. The second experiment shows another target moving across the scene with a range of several hundred feet.

Figure 17: In the first experiment, MATS identifies the target location in the map and highlights the vehicle's position.

Figure 18: MATS generates a side view of the first scene showing what the vehicle should expect to observe and highlights the target in red, which corresponds very well to the location of the target in the actual images.

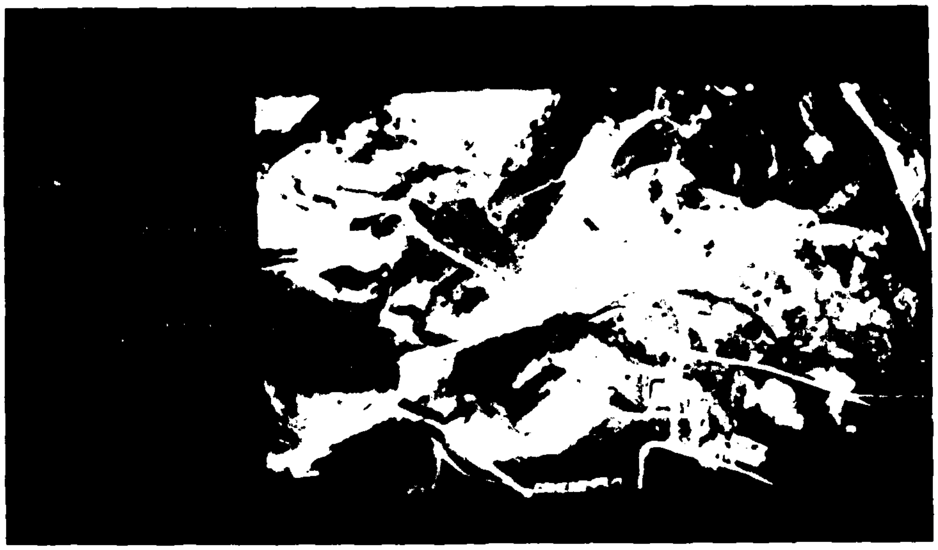


Figure 19. Target location detected in the second experiment with a direction vector overlaid on the target indicating its traveling course.

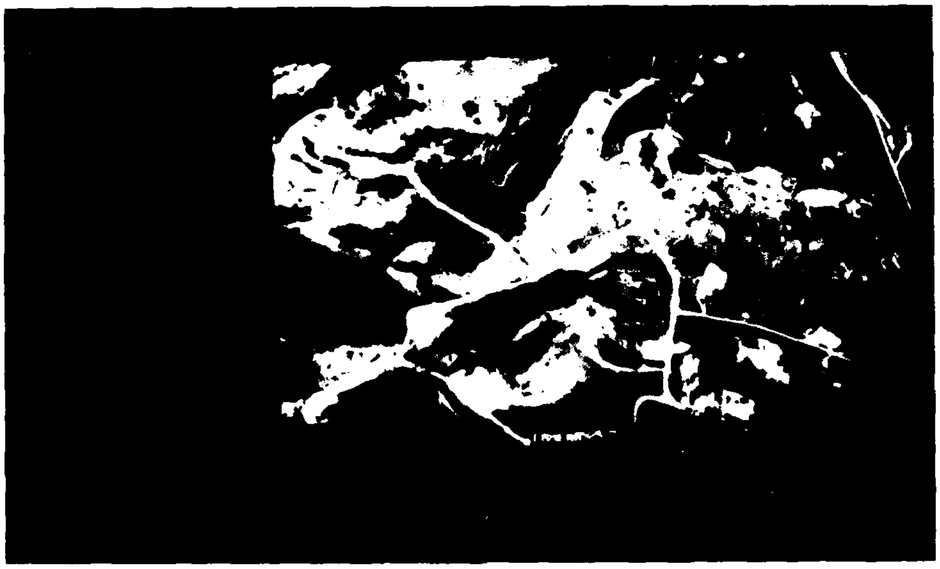


Figure 20. Final results of the second experiment which highlight the road segment the target is traveling on.

The objective of the MATS effort is to demonstrate the practical aspects of target tracking in a real-world scenario, where scenes are usually highly cluttered, low in contrast, and contain targets at long ranges. MATS has shown that it can enhance target detection and tracking performance and provide useful outputs such as a target's 3-D map location and the road it is traveling on. Currently, MATS is still a prototype system which needs further development. More experiments are planned to establish the performance capabilities of this system.

7. REGION MOTION DETECTION USING COLOR DIFFERENCE PICTURES

When the 3-D world is projected onto a 2-D image, valuable information is lost. Motion information, along with stereopsis and range maps, is a well known information source for the reconstruction of 3-D representations from 2-D images. Since motion information is useful for other image processing stages, it is desirable to perform motion analysis at an *early* stage of scene understanding. This early stage is referred to as the *peripheral process* by Jain[14] and the *short-range process* by Ullman.[23] The long range or attentive processes are correspondence schemes in which high-level, symbolic features are matched and tracked over time. Low-level processing stages for motion interpretation include gradient-based methods,[13] cross-correlation methods,[1] and spatio-temporal filtering methods.[24]

For real-time motion analysis, the algorithms employed are constrained to be efficient as well as dependable. Jain[14] has experimented with a simple method of using a difference picture accompanied with a simple decision tree to extract motion information in the peripheral phase. A difference picture is generated by comparing two frames of the same dynamic scene on a point by point basis. In subsequent experiments, he showed that the difference picture, in combination with the edge and corner image, could be used effectively to detect motion in the scene.[14,16] Features such as temporal-edges and interest points are often used in motion detection algorithms. Examples are the edge features used by Hildreth[12] to compute optical flow and the region images used by Bhanu and Burger[4] to compute disparity vectors. However, one deficiency of the Jain approach is that the interiors of constant intensity level regions do not generate a difference signal, even if the corresponding surface is moving. Thus, the determination of surface boundaries requires the observation of longer sequences or the application of more sophisticated, high-level analysis.

We present a similar scheme using color images to obtain a more reliable difference picture for use with standard region-based motion detection schemes. We have successfully demonstrated the detection of motion for the complex ALV imagery, where Jain's algorithm normally is not robust enough due to the diverse nature of this imagery.

7.1 COLOR DIFFERENCE PICTURE

Motion analysis techniques use various assumptions about the scene characteristics to decrease the complexity of the calculations of 3-D features. One such assumption is that the illumination of an object does not change from scene to scene. For the ALV scenario, this assumption may not hold because the changing location of the imaging system causes the orientation and location of objects, relative to the ALV, to continually change. Thus, the use of invariant scene characteristics is necessary. It has been reported that changes in the ambient illumination level does not alter the human perception of color. By using the individual color components of the image, instead of the luminance component, gradient-based motion algorithms will be less sensitive to local changes in average object intensities. For example, the hue of the image is calculated as a function of a ratio of linear combinations of the three primary image intensities, red, green, and blue.

$$Hue = \cos^{-1}\left\{ \frac{\tfrac{1}{2}[(R-G)+(R-B)]}{\sqrt{(R-G)^2+(R-B)(G-B)}} \right\} \tag{16}$$

If $B > G$, then $Hue = 2\pi - Hue$

Therefore, a change in the average intensity of a specific region does not effect the magnitude of the hue.

The temporal derivative $\frac{\partial f(x,y,t)}{\partial t}$ in the discrete domain is approximated by the difference operation $f(x,y,t_2) - f(x,y,t_1)$, where $f(x,y,t)$ is the image intensity at the location $\{x,y\}$, at time t. The entries in the difference picture are significant only at pixel locations where an object has moved. The difference picture may be used in combination with the edge image to obtain a time-varying edge detector.[16] When more than two frames are used to approximate the temporal derivative, observation of a time-varying edge permits the detection of moving edges.[16] This distinction is made since the motion of an object detected with a Moving Edge Detector results from *persistent change* in the sequence of the frames representing the scene. Jain argues that if the change exists only in two frames, then the change is, most probably, not due to motion. The use of multiple frames helps to resolve the ambiguity problems due to noise that occur for frame-to-frame differencing techniques.

In the Moving Edge Detection method, the difference picture is refined using a syntactic labeling scheme. Because the criteria for this scheme are derived for noiseless imagery, they work accurately when there are only minor changes in the average intensity level of regions and the edges of regions are sharp. We have developed a similar, but simpler, method of obtaining a difference picture which uses color images in order to make the algorithm less sensitive to these inconsistencies.

Our algorithm is comprised of three steps:

(1) The point by point subtraction is done for each primary image, resulting in three conventional difference pictures.

(2) A symbolic mapping is applied to the red, green, and blue difference values to obtain a single symbolic difference image. (See Table 1). Only the sign of the difference picture for each primary image is used to derive the symbolic difference image, because *how* the region changed is more important than *how much* it changed.

(3) The connected components of the symbolic difference image are extracted. Isolated points and small regions are eliminated. Small regions in the difference picture are either caused by noise or by objects that are too far from the ALV to be of interest.

The computational complexity of this algorithm is low. Since the change in the region intensity, rather than the magnitude of the change, for each of the three primary images is used for moving object detection, the scheme is robust for outdoor imagery, when the camera rotation component is not significant. This procedure effectively removes most of the noise that often exists in difference pictures. Only those regions which are caused by targets moving at a *significant rate* of speed remain. The results obtained with this approach demonstrated far less noise than the difference pictures obtained from the Jain algorithm alone.

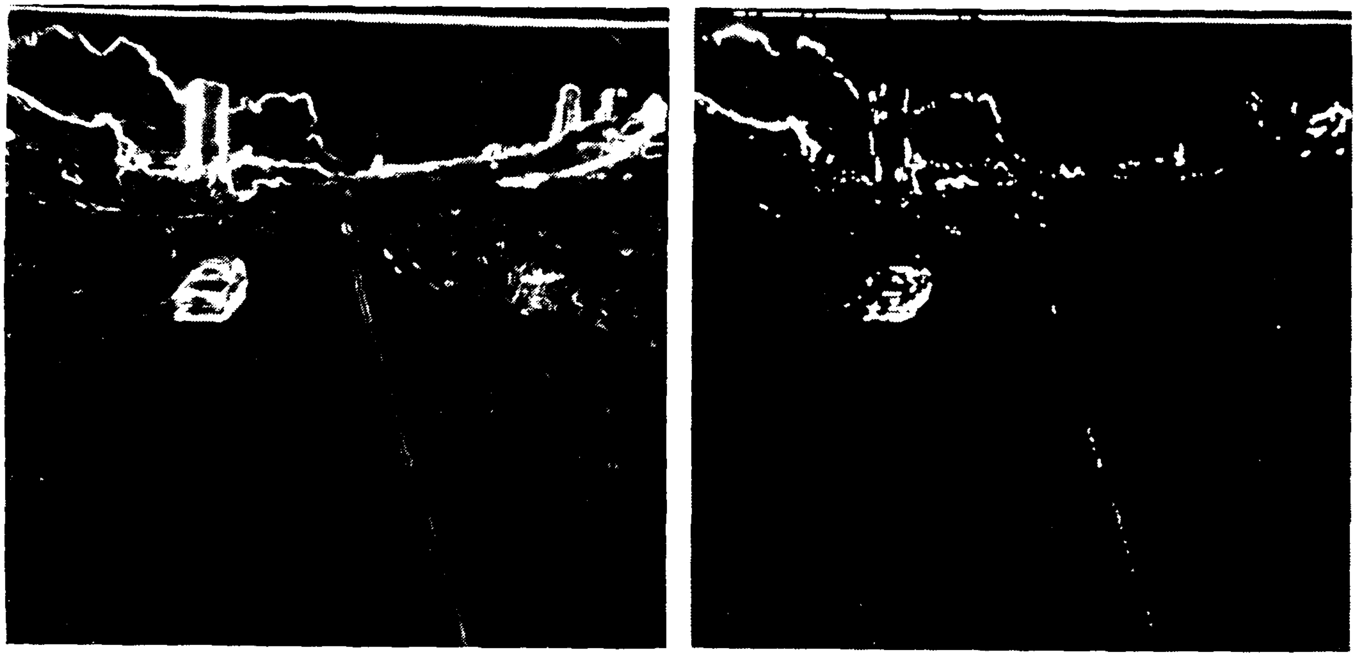
The performance of the Color Difference Picture motion detection algorithm was evaluated for a representative set of dynamic images. The purpose of the evaluation was to empirically appraise the effects of changing the algorithm's parameters on the resultant number of false detections, for imagery with varying amounts of sensor-induced motion. It was found that the approach performs well for sequences of images where the imaging system's motion is approximately linear (a large forward component, plus a significantly smaller rotational component), with only moderate sensitivity to the selected parameter values. For image sequences demonstrating more complex motion (significant rotational components), the algorithm was more dependent on the parameter values selected and it was not possible to identify a set of parameters for the algorithms which would be optimal for all cases.

7.2 MOVING COLOR EDGE DETECTION

Moving targets can be detected in a series of images on the basis of multiple clues. A few examples are moving edges, moving corners, and spatio-temporal frequency disparities. For Moving Color Edge Detection, the color difference picture is combined with the color edge magnitude image to identify moving color edge points in a sequence of images. This technique detects image locations where there is a high degree of *edginess* and a high temporal rate of change in intensity. Regions that display a large temporal rate of change are those regions of the color difference picture that remain after small regions are discarded. The edge image is obtained with the DiZenzo color image edge operator.[11] The magnitude of the DiZenzo color image edge operator for a scene where a vehicle is rapidly approaching the ALV is presented in Figure 21. The result of deriving the conjunctive evidence of the multi-image gradient magnitude and the Color Difference Picture is presented in Figure 21(b). Figure 22 provides another example of Moving Color Edge Detection.

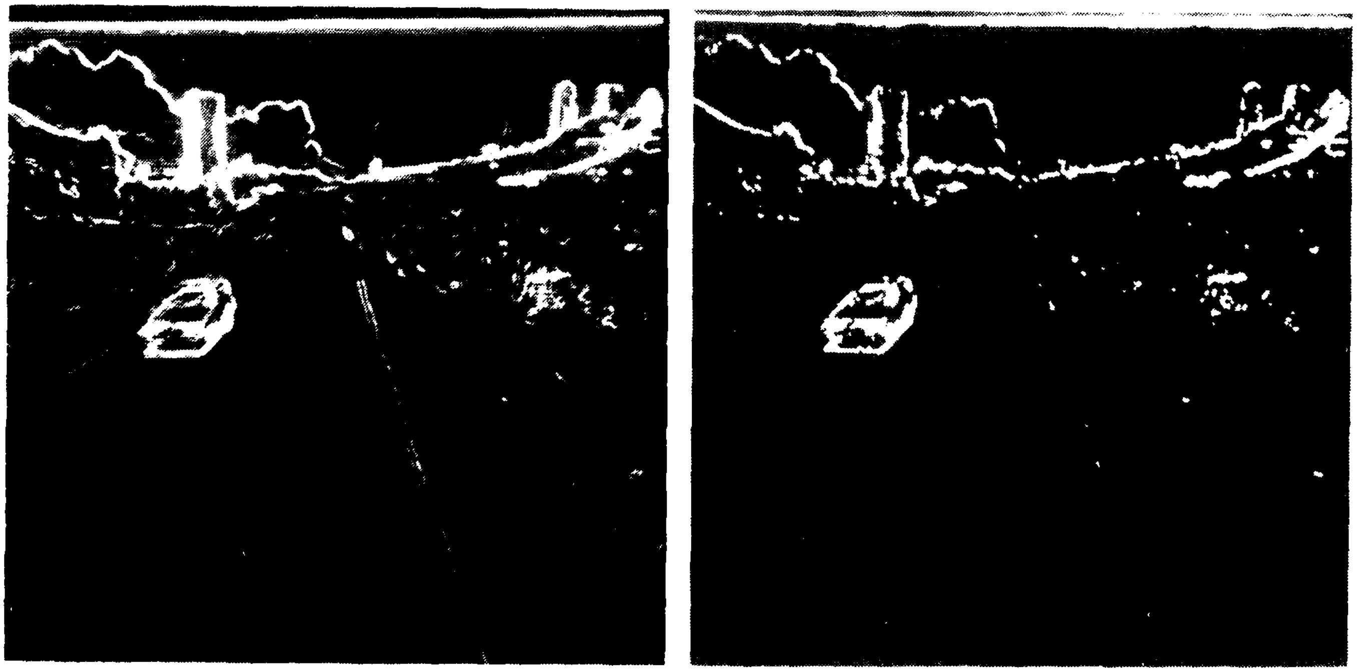
Class	Red	Green	Blue	Meaning
0	0	0	0	No change or changed negatively in all colors
1	0	0	1	changed positively in Blue direction
2	0	1	0	changed positively in Green direction
3	0	1	1	changed positively in Blue and Green directions
4	1	0	0	.
5	1	0	1	.
6	1	1	0	.
7	1	1	1	positive change in all three colors

Table 1: Symbolic mapping of three color difference images. 1 in column R, G, and B indicates a positive change.



(a) *(b)*

Figure 21 Color Difference Picture Motion Detection results. *(a)* Magnitude of the gradient image calculated with the DiZenzo Color Edge Detection Operator for frame 40. *(b)* Moving color edges detected in frame 40.



(a) *(b)*

Figure 22 Color Difference Picture Motion Detection results. (a) Magnitude of the gradient image calculated with the DiZenzo Color Edge Detection Operator for frame 44. *(b)* Moving color edges detected in frame 44.

7.3 MOVING REGION DETECTION

For ALV imagery, motion may be detected as a result of either true object motion or apparent (sensor-induced) motion because the imaging system is constantly translating and rotating due to the undulating road surface. The performance of the Moving Edge Detection operator and other traditional techniques will deteriorate for this variety of imagery unless a mechanism is used to compensate for the motion of the imaging system.

Conventional background motion compensation techniques match significant image features to obtain an estimate of the translation and rotation of the imaging system from the previous frame to the current frame. The transformation must be estimated based on a minimum mean-square error criterion between the observed location of the feature points and the predicted location, on the basis of the estimated transformation. The calculation of this transformation can be very expensive. Therefore, motion detection algorithms that are designed for a moving camera and that don't compensate for sensor-induced motion must be very robust.

We derived such an algorithm by modifying Jain's approach for greater resistance to the detection of changes caused by sensor motion and not by target motion. The Moving Boundary Detection algorithm detects those regions of the image whose location is changing rapidly with the following principle: Targets seen in an image will demonstrate a significantly higher rate of translation in proportion to their size than an arbitrary background region will, due to sensor motion.

We define a rate measure using this principle. First, the input color image is partitioned into a set of connected regions employing a region-based segmentation algorithm such as the Ohlander-Price-Reddy algorithm.[19] The regions where significant motion occurs are obtained by masking the regions detected with the Color Difference Picture (CDP). The criterion function for each region is calculated as the ratio of the number of boundary points which changed location to the total number of boundary points:

$$rate = \frac{\text{\# pixels of boundary that changed location}}{\text{total boundary length}} \tag{17}$$

This technique will exhibit degraded performance if any one of the following conditions apply: 1) the distance at which moving objects must be detected is extremely large, in which case the objects appear as part of the background; 2) the moving objects are small; or 3) the objects are moving at a sufficiently slow rate of speed, so that little or no change in the object's location is detectable at 30 Hz. Because none of the preceding degenerate cases occurs for the ALV scenario, the Moving Boundary Detection algorithm is a good algorithm for change detection.

This criterion function is useful for discriminating moving targets from stationary targets and/or the background for the following reasons: If image disparities for a pair of frames are the result of imaging system rotation only, then the apparent motion imparted to distant portions of the background will be greater than the apparent motion of stationary objects in the near-field. The spatial resolution of the image of an object is inversely proportional to its range from the imaging system. Thus, because the resolution of the background is low, its segmentation is poorer than the segmentation of nearer portions of the scene. Therefore, these regions of the image will be eliminated by the region size criteria in the formation of the CDP. The regions of the background that aren't discarded by the preceding step are normally segmented out by the rate measure, because they translate at a relatively slow rate. The only regions of the image that remain after these two segmentation phases are the moving objects. The range of interest for object motion detection will dictate the thresholds for this process. Results obtained with this approach for four frames of the Collage I database are presented in Figure 23.

A Moving Region Detection algorithm was also implemented. This algorithm is derived from the same concepts that applied for the derivation of the Moving Boundary Detection algorithm. Its "rate" measure is:

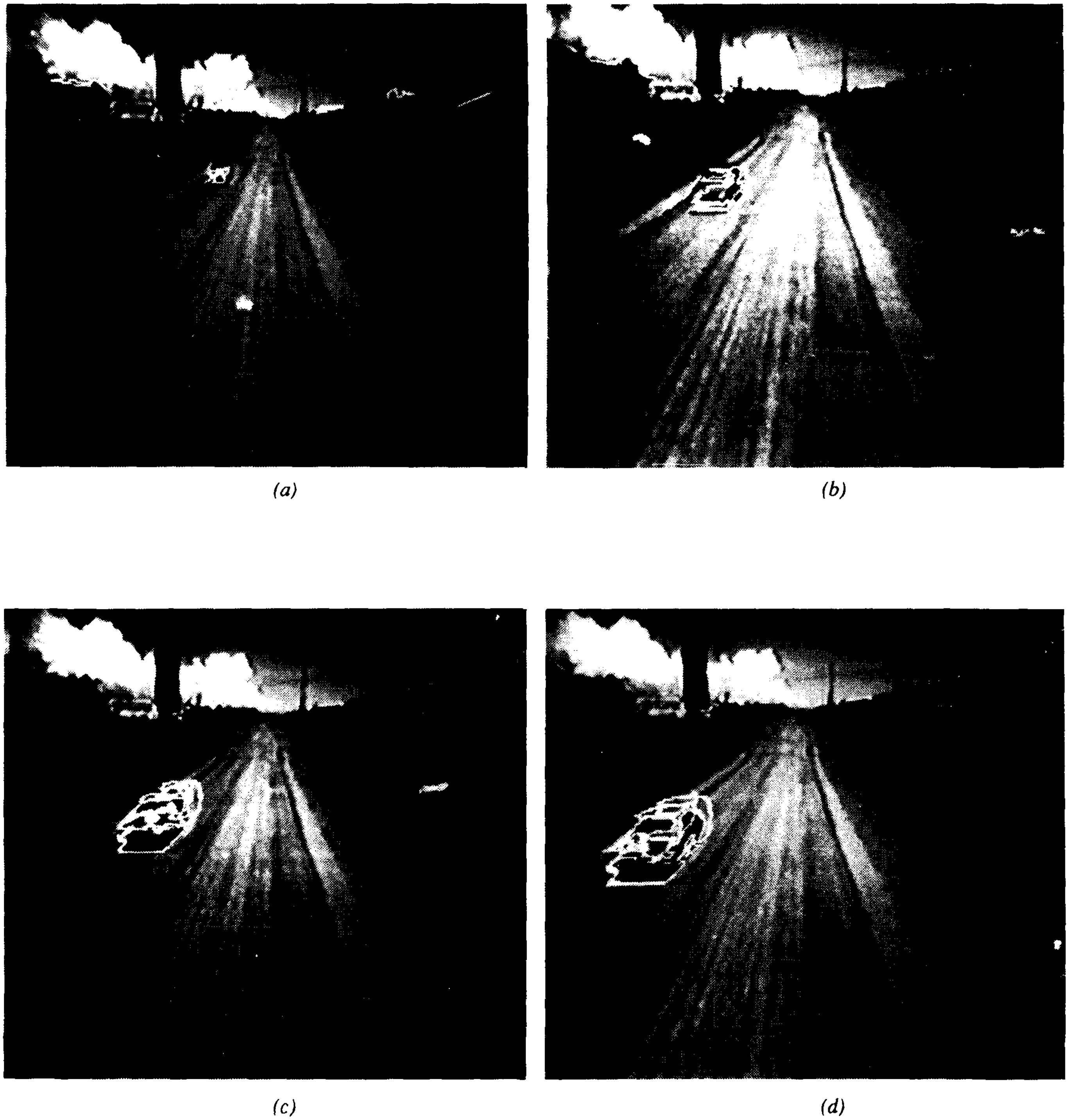
$$rate = \frac{\text{\# pixels that moved}}{\text{total area of the region}} \tag{18}$$

The performance of this algorithm was comparable to the performance of the Moving Boundary Detection algorithm. The regions which resulted after application of this algorithm to the images are shown in Figure 24.

8. CONCLUSIONS

We have presented our qualitative approach to scene understanding for mobile robots in dynamic environments. The challenge of understanding unstructured outdoor image sequences is that stationary objects do not appear as stationary in the image and mobile objects do not necessarily appear to be in motion. Consequently, the detection of 3-D motion often requires reasoning far beyond simple 2-D change analysis.

The approach taken here clearly departs from related work by following a strategy of qualitative, rather than quantitative, reasoning and modeling. All the numerical efforts are packed into the computation of the Focus of Expansion (FOE), which is accomplished entirely in 2-D. To cope with the problems of noise and errors in the



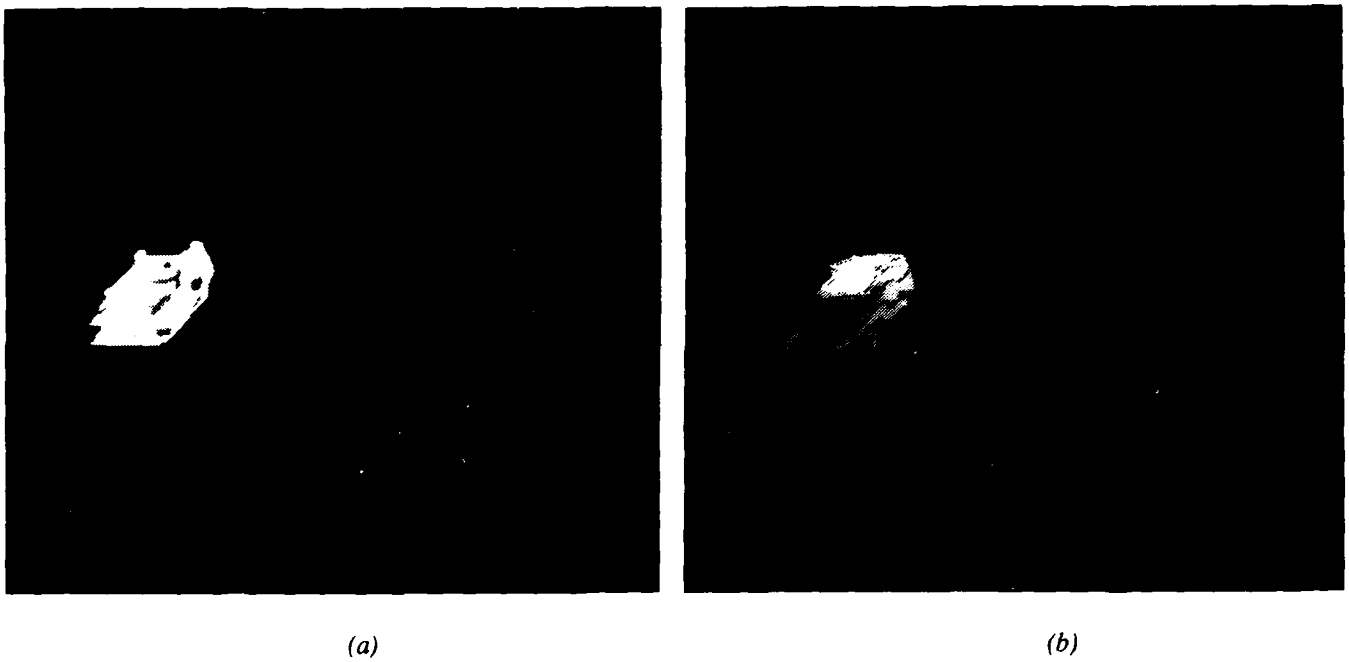
(a)

(b)

(c)

(d)

Figure 23: Moving Boundary Detection algorithm results. The boundary of the detected moving target is shown with a white line. *(a)* Frame 20. *(b)* Frame 40. *(c)* Frame 44. *(d)* Frame 48.



(a) (b)

Figure 24: Moving Region Detection algorithm results. The detected moving regions are displayed as uniform intensity regions, where the intensity of each region is arbitrary. Non-moving regions are displayed as black. *(a)* Frame 48. *(b)* Frame 50.

displacement field, we determine a region of possible FOE-locations, known as the *Fuzzy FOE*, instead of a single FOE.

We have shown that even without knowing the exact location of the FOE, conclusions about motion and 3-D scene structure can be drawn. From these clues, we construct and maintain an internal 3-D representation, termed the *Qualitative Scene Model*, in a generate-and-test cycle over extended image sequences. This model also serves as a platform for other visual processes, such as occlusion analysis, perceptual grouping, and object recognition. To overcome the ambiguities inherent to dynamic scene analysis, multiple interpretations of the scene are pursued simultaneously.

The examples given in this paper show the fundamental operation of our approach on real images produced by the Autonomous Land Vehicle (ALV). Since the exclusive use of displacement vectors from point features is a limiting factor, we showed our initial experiments on edge and region-based feature tracking. We plan to integrate wavefront approaches also for region motion detection.[4] Also, to exploit a larger part of the information contained in the image and to demonstrate the full potential of our approach, lines, regions, and map information need to be fully integrated within the Qualitative Scene Model.

ACKNOWLEDGEMENTS

We would like to thank George Lukes of the U.S. Army Engineer Topographic Laboratories for his input on the map-related experiments. We are also thankful to Dave Robohm and Henry Lai for their help in this work. This research is supported by DARPA/U.S. Army ETL Contract DACA 76-86-C-0017.

REFERENCES

1. S.T. Barnard and W.B. Thompson, "Disparity Analysis of Images," *IEEE Trans. on Pattern Analysis and Machine Intelligence* **PAMI-2**(4) pp. 333-340 (July 1980).
2. B. Bhanu, "Automatic Target Recognition: State of the Art Survey," *IEEE Trans. on Aerospace & Electronic Systems* **AES-22**(4) pp. 364-379 (July 1986).
3. B. Bhanu and W. Burger, "Scene Dynamics Technical Report - DRIVE: Dynamic Reasoning from Integrated Visual Evidence," Defense Advanced Research Projects Agency, Contract No. DACA76-86-C-0017, Honeywell Systems and Research Center (June 1987).
4. B. Bhanu and W. Burger, "Approximation of Displacement Field Using Wavefront Region Growing," *Computer Vision, Graphics and Image Processing*, (March 1988).
5. B. Bhanu and W. Burger, "Qualitative Motion Detection and Tracking of Targets from a Mobile Platform," *Proc. DARPA Image Understanding Workshop*, pp. 289-318 (April, 1988).
6. B. Bhanu and D. Panda, "Qualitative Reasoning and Modeling for Robust Target Tracking and Recognition from a Mobile Platform," *Proc. DARPA Image Understanding Workshop*, pp. 96-102 Morgan Kaufmann, (1988).
7. W. Burger and B. Bhanu, "Qualitative Motion Understanding," *Proc. Tenth International Joint Conference on Artificial Intelligence, IJCAI-87, Milan, Italy*, Morgan Kaufmann Publishers, (August 1987).
8. W. Burger and B. Bhanu, "Qualitative Understanding of Scene Dynamics for Autonomous Mobile Robots," *Submitted to International Journal of Robotics Research*, (1987).
9. W. Burger and B. Bhanu, "Dynamic Scene Understanding for Autonomous Mobile Robots," Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (June 1988).
10. W. Burger and B. Bhanu, "On Computing a 'Fuzzy' Focus of Expansion for Autonomous Navigation," Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (June 1989).
11. S. DiZenzo, "A Note on the Gradient of a Multi-Image," *Computer Vision, Graphics and Image Processing* 33(1) pp. 116-125 (January, 1986).
12. E.C. Hildreth, *The Measurement of Visual Motion*, MIT Press, Cambridge, Mass. (1984).
13. B.K.P. Horn and B.G. Schunck, "Determining Optical Flow," *Artificial Intelligence* **17** pp. 185-203 (1981).
14. R. Jain, "Extraction of Motion Information from Peripheral Processes," *IEEE Transactions on Pattern Analysis and Machine Intelligence* **PAMI-3**(5) pp. 489-503 (1981).
15. R. Jain, "Direct Computation of the Focus of Expansion," *IEEE Transactions on Pattern Analysis and Machine Intelligence* **PAMI-5**(1) pp. 58-64 (January 1983).
16. R. Jain and S. Haynes, "Time-varying Edge Detection," *Computer Graphics and Image Processing* **21** pp. 345-367 (1982).
17. J. Kim and B. Bhanu, "Motion Disparity Analysis Using Adaptive Windows," Technical Report 87SRC38, Honeywell Systems & Research Center (June 1987).
18. H.P. Moravec, "Towards Automatic Visual Obstacle Avoidance," Proc. 5th International Joint Conference on Artificial Intelligence, pp. 584 (August 1977).
19. R. Ohlander, K. Price, and D.R. Reddy, "Picture Segmentation Using a Recursive Region Splitting Method," *Computer Graphics and Image Processing* **8** pp. 313-333 (1978).
20. K. Prazdny, "Determining the Instantaneous Direction of Motion from Optical Flow Generated by a Curvilinear Moving Observer," *Computer Graphics and Image Processing* **17** pp. 238-248 (1981).
21. D. Regan, K. Beverly, and M. Cynader, "The Visual Perception of Motion in Depth," *Scientific American*, pp. 136-151 (July 1979).
22. J.H. Rieger, "Information in Optical Flows Induced by Curved Paths of Observation," *J. Opt. Soc. Am.* 73(3) pp. 339-344 (March 1983).
23. S. Ullman, *The Interpretation of Visual Motion*, MIT Press, Cambridge, Mass. (1979).
24. A. B. Watson and A. J. Ahumada, Jr., "Model of Human Visual-Motion Sensing," *Journal of the Optical Society of America A*, 2, pp. 322-342 (1984).

DESCRIPTIONS OF COMPLEX OBJECTS FROM INCOMPLETE AND IMPERFECT DATA*

Kashipati Rao† and Ramakant Nevatia
Institute for Robotics and Intelligent Systems,
Powell Hall 204, MC-0273,
University of Southern California,
Los Angeles, CA-90089.

ABSTRACT

Usually shape description systems assume that a scene has been segmented into objects and that object boundaries are given. However, this is not realistic when working with intensity images; the resulting boundaries are fragmented and also contain surface markings, and shadow and noise boundaries. Our system works with such input and computes shape descriptions of complex objects. Scene segmentation takes place through the process of shape description. We use generalized cones, or more precisely, their 2-D analogs of ribbons as the basic shape representation scheme. We show results on several synthetic and real examples. The output of our system should be useful for object recognition and for further inference of 3-D shape from the 2-D shape descriptions.

1 INTRODUCTION

Computing descriptions of shapes of objects in a scene is one of the most central problems in machine vision. Good shape description is needed for object recognition, of course, but also for other tasks requiring geometric reasoning about the scene, such as grasping or navigation.

Normally, shape description is assumed to be preceded by a *scene* segmentation process that outlines different objects in the scene, *i.e.* the "figure-ground" resolution problem is assumed to be solved. While this is possible given high quality, dense range data, it is usually not so for realistic intensity images as the image boundaries are likely to be incomplete and imperfect, containing boundaries arising from markings, shadows and noise in addition to the object boundaries (see Figure 1 for example.) In such cases, the object shape itself is helpful in achieving good segmentation, *i.e.* scene segmentation and shape description are inter-dependent. In our system, we do not assume that we know the specific shapes that we expect to perceive, but only certain generic classes of shapes as described below.

We have chosen a segmented, hierarchical representation for our shape descriptions. In this approach, a scene is represented in terms of component objects; complex objects are represented by decomposing them into simpler parts and then describing the parts and relations between them. This process can be applied hierarchically. Such representations can be rich and stable, and represent occlusion and articulation in a natural way.

In this work, we have chosen *generalized cones* for the basic part representation. A generalized cone (GC) consists of an arbitrary planar shape, called a *cross-section* swept along an arbitrary 3-D curve, calleo an *axis*. Further, the size and also the shape of the cross-section may change along the axis; the rule describing the change is called the *cross-section function*. We can also define 2-D analogs of GCs, often called *ribbons*. For a ribbon, the axis is an arbitrary 2-D curve, the "cross-sections" are simply line segments normal to the axis and the cross-section function defines how the cross-section width changes along the axis. 2-D ribbons may be viewed as projections of 3-D GCs. Given only a single intensity image, it is easier to compute the ribbons in the image which may serve as a step towards computing the 3-D volume descriptions.

GCs have been used for shape description for several years and by several researchers. However, most of that work assumes that the input consists of a "perfect line drawing", *i.e.* the scene has been segmented into objects and that the extremal object contours have been made explicit [9, 6]. Such input can be expected if dense range data

*This research was supported by the Defense Advanced Research Projects Agency under contract number F33615-87-C-1436, monitored by the Air Force Wright Aeronautical Laboratories, Darpa Order No. 3119.

†Now with: Image Understanding Branch, Computer Science Center, Texas Instruments, P.O. Box 655474, MS 238, Dallas, Texas 75265

is available; even then, the object surfaces are typically prepared specially, such as being painted mat white. Use of intensity images invariably gives fragmented boundaries, and we must distinguish object contours from contours caused by shadows, surface markings and noise. Some typical scene boundaries are given in Figure 1 (note that all but (b) are from real images). The *ACRONYM* system does deal with intensity image inputs and fragmented boundaries but is highly model-directed [3].

In [11] we described a system that can compute GC descriptions from sparse, imperfect data. This system assumed that the input consisted of boundary fragments, but that boundaries were given in 3-D (as may be obtained from stereo, for example). This system is limited to the domain of the "Linear, Straight, Homogeneous, Generalized Cones" (LSHGCs) (in Shafer's terminology [12]).

This paper describes a much more general system. We assume that the scene consists of objects that are well described as GCs (or as ribbons in the image), but the the GCs can be general, *i.e.* they may have curved axes and the cross-sections need not change linearly. Furthermore, the objects may be compound objects, *i.e.* be composed of simpler GCs or other compound objects. The system has no *a priori* knowledge of the specific shapes it expects to see (other than that they should be well described by GCs). Figure 1 shows typical examples of such a scene. The scenes are deceptively simple for humans but handling them requires several capabilities: the figure/ground problem needs to be resolved, the object boundaries need to be made explicit from other boundaries, and finally a shape description needs to be computed. Note that simple "axis-finding" techniques such as those of Blum [1] and Brady [2] cannot be applied to such figures as we do not have closed boundaries.

As we use only a single image, our system produces ribbon descriptions only, and not the 3-D GC descriptions. The 3-D GC description would require an additional step of inferring 3-D structure from the 2-D descriptions, either from the monocular information itself or from one or more other views. We believe that the ribbon description is precisely the right one for inferring 3-D shape from 2-D shape (as shown in [13]). Similarly, we also believe that the ribbon description will aid in stereo analysis (use of high level descriptions in stereo analysis has been demonstrated in [5, 7], for example).

In the next section, we briefly describe our approach to this problem and then give details in sections 3 and 4. Several experimental results are shown in section 5.

2 APPROACH AND OVERVIEW

Our approach is based on the following *very basic* and *fundamental* observation:

Finiteness Observation O1:

Objects/object parts are not infinite in extent, they are terminated either by a boundary or another object part.

This observation is embodied in the following principle:

Principle of Termination P1:

If the boundaries do define a ribbon, the ribbon must "terminate" somewhere. If the ribbon represents the entire object, then it must be terminated at both extremities by terminator boundaries. If it is part of an object, it may be "terminated" at either (or both) ends by another part.

This rather obvious observation leads to a paradigm which we hope to show is very powerful. As the system makes no other assumptions, we need to put no other restrictions on the scene and need no *a priori* knowledge of the specific shapes in the scene.

At the top-level, our approach is simply that of "hypothesize and verify" (see Figure 2). Hypotheses generation consists of finding some group of boundaries that may define an object or parts of it. Verification consists of applying additional criteria to establish which hypotheses are viable. In our approach, hypotheses formation is essentially that of finding boundary segments that can define a ribbon. Some ribbons are preferred over others, and the preference is determined by the shape of the axis and the cross-section function. Verification criterion is based on the **Finiteness Observation O1** and the **Principle of Termination P1**.

Even though, the conceptual approach is simple, its implementation is not and requires sophisticated *geometric reasoning*. The low-level process of finding ribbons can find a very large number of candidates (for the two hammers example in Figure 1 (b) there are 7240 axis points, 872 ribbons and 261 super-ribbons or larger ribbons — some of these terms are defined accurately later), a vast majority of which we humans would find very unnatural and do not even seem to consider in our perception (this again shows the pitfalls of introspection as a means of understanding human perception). We must now determine what the geometric relations between these candidates are and which candidates are supported by other evidence as being possible objects.

Our verification procedure is based on the following key ideas (to be elaborated on in section 4):

Same-sidedness Observation O2:

Two ribbons connected by a boundary and lying on opposite sides of it cannot both be simultaneously acceptable:

one of the ribbons has to be in the figure and the other in the background. That is, the boundary is an *occluding boundary* between the two ribbons. (This observation is true except for accidental alignments.)

Figure 3 explains this observation. Consider three ribbons R1, R2 and R3 in the picture. Consider the *ribbon-inbetween boundary* b_r joining R1 and R2. [It is the list of edgels (or edgelts) joining the extremity points of R1 and R2 as shown in the figure.] For convenience let b_r also be the ribbon-inbetween boundary joining R1 to R3. R1 and R2 lie on the same side of b_r and are, therefore, acceptable as being together in the figure or in the background. R1 and R3 lie on opposite sides of b_r and cannot both be accepted simultaneously: one of them has to be in the figure and the other in the background.

Non-overlapping Observation O3:
This observation has the following ideas:

1. The joint formed by ribbons does not share area with the composing ribbons. This is illustrated in Figure 4.
2. Two ribbons participating in a joint do not share any area (they may share boundary edges though). (This is shown schematically in Figure 10 (d).)

Note that in our approach scene segmentation and object description are not separate steps. Traditionally, one first segments a scene into objects and then derives shape descriptions for the segmented regions. In our approach, segmentation is a by-product of having found the desired shapes in the scene!

Our approach in this system (*SHAPE II*) is similar, conceptually, to that in our earlier system (*SHAPE I*) for LSHGCs. However, hypotheses for LSHGCs can be highly constrained; we were able to use the property that the contour generators are single linear segments and must be co-planar, and the verification process was much simpler. In the current system, we must accept many more hypotheses, and the verification needs to do much more reasoning.

3 HYPOTHESES GENERATION

Hypotheses generation essentially consists of finding boundary fragments that form acceptable ribbons. A ribbon is defined by its axis and "cross-sections" which are merely line segments. We require the cross-section to be orthogonal to the local tangent of the axis. Figure 5 gives the block diagram of the hypotheses generation part of the *SHAPE II* system.

The first step consists of finding pairings of boundary edges in various directions that could give rise to axis points of ribbons. The axis points are linked into *axis-threads* based on contiguity. Then, axis points that do not come from orthogonal cross-sections are eliminated. We now have fragments of ribbons which are linked into *super-ribbons.* Super-ribbons, with associated descriptions, are the output of the hypotheses formation stage. We now describe each step in a little more detail.

The input to the system is a line drawing or an edgel image, i.e., it has 1's where there are edgels and 0's elsewhere. In general, this image may have breaks in the boundaries of the objects and there may be surface marks on the objects and the background. There may be shadows too. We use linked edgels from such an image.

Finding axis points: The first step is to find potential axis points. We do this by finding potential axes in various directions. To find axes in the horizontal direction, for example, we "draw" vertical lines (or strips) in this image at consecutive pixels (or at fixed intervals) and find the edgels on each such vertical line. Axis points are then placed mid-way between every pair of edgels in the line. Similarly, axis points are found in other directions. This is done by rotating the image, or merely, the edgel list and drawing the "vertical" lines again. We find axis points in some equally spaced directions between 0 to 180°. (Our method of finding these axis points is basically the method of projections suggested by Nevatia and Binford [9].) Figure 6 explains some of the terminology used. (Some of the terms illustrated in the figure are defined and used later.)

An axis point has a data structure associated with it. It consists of the x (column) and y (row) location of the axis point, the two edgels defining the axis point (called the *boundary points*), the distance between the edgels, the angle of rotation of the image (from which the axis point came) and the intermediate pixels between the boundary points (if necessary). (We call the axis point with its associated data structure a *bead.* It may also be understood as a *ribbon point* in the context of the ribbons discussed later.)

Linking the beads: For each direction, we order and group the beads by linking them into "ribbons" or 2-D generalized cones (alternatively, the axis points are linked into "axis-threads") by a near-neighbor algorithm. The principle behind the linking and the neighbor checking operation is:

Principle for Linking Beads P2: *linked edgels in the image give rise to linked beads.*

Filtering out deviations: The above procedure of linking gives us several axis-threads. However, at this stage, we cannot guarantee that the axis is always orthogonal to the cross-sections it came from. Those points where the local tangent at the axis is not nearly perpendicular to the vertical strips or vertical lines used to obtain the axis points are filtered out.

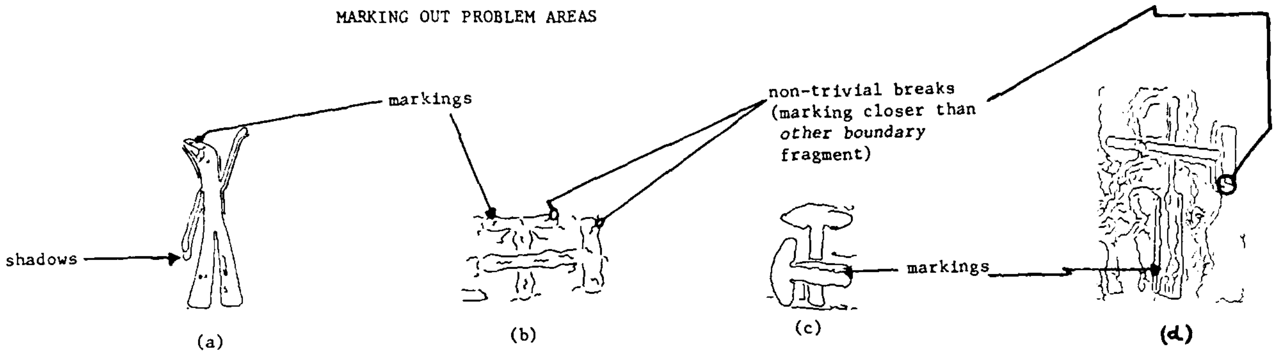


Figure 1: Finding the axes of compound objects in scenes like these with breaks in the boundaries, surface marks and occlusion is, in general, a difficult problem.

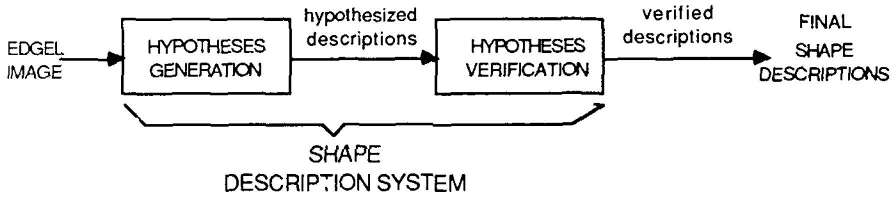


Figure 2: Block diagram of the SHAPE II system

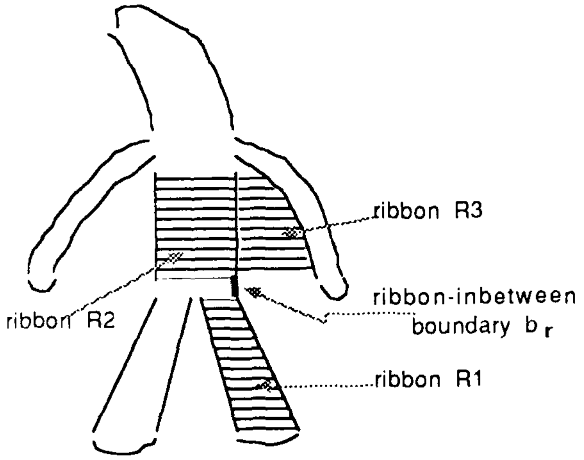


Figure 3: The same sidedness of a curve observation

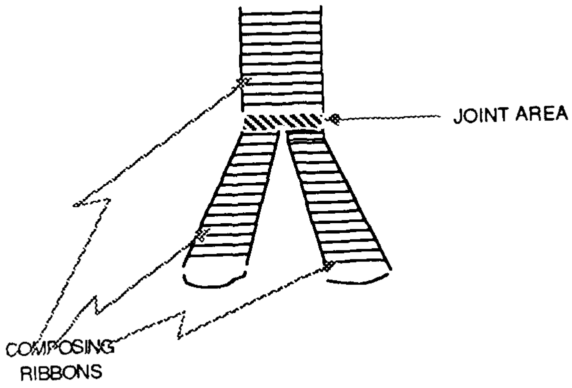


Figure 4: The joint area should be disjoint from the areas of the composing ribbons

Making and rating ribbons: Each axis-thread defines a 2-D generalized cone or "ribbon". Ribbons are rated by length-to-width ratio, uniformity of width and uniformity of curvature of their axes. These ratings are useful but *not* critical to the performance of our system.

Forming and rating super-ribbons: We join ribbons obtained from the previous stage into "super-ribbons" by contiguity. We then rate super-ribbons just as we rate ribbons. (Here again, these ratings are useful but *not* critical to the system.) The criteria for contiguity between two ribbons at the extremities being considered are: at those extremities the ribbons should be nearby; the tangents to their axes and their boundaries at those extremities should be continuous; the ribbons should have come from neighboring angles of rotation of the image (for obtaining the axes); the widths of the ribbons at those extremities should be continuous; and one ribbon should not be completely contained in the other or *vice versa*.

We conclude this section by referring to the result of hypotheses generation on the two hammers example. Figure 18 (b) shows the large number of hypotheses (261 super-ribbons) generated, most of them non-intuitive. (For clarity, we just show the axes of the super-ribbons rather than their regions.) In the next section we discuss the verification stage, which is used to choose from among these large number of hypotheses.

4 HYPOTHESES VERIFICATION

The hypotheses generation phase gives rise to a rather large number of hypotheses, most of which do not correspond to the objects we perceive. We now choose among these based on a *verification* phase. The basic concept is that the hypotheses generation stage considered only the boundary fragments that defined the axes. However, if the ribbon does correspond to an object (or its part), it must also have ends or attach to another object (part). Figure 7 gives the block diagram of the hypotheses verification stage.

As the figure shows, the verification module proceeds by first checking extremities of all ribbons for *terminator* boundaries. If a ribbon is terminated (the word terminated will now on be used in the sense of being terminated by a boundary) at both extremities, it is completely verified and declared an object. (It is a simple object, with no parts, as opposed to a compound object.) For the rest of the ribbons, we check if they form possible joints with one another and if the joints could themselves be grouped to describe a compound object, which is verified at all its extremities. This method of grouping ribbons into joints is performed by forming a graph of the ribbons, pruning out the ribbons without a potential to be verified and then looking for cycles in the graph — which are joints in the object. The ribbons in the cycles are then merged, if possible, by continuity. The cycles are grouped into larger cycles (called supercycles) and the extremities of the supercycles are checked if they are verified. This process may be understood as a method of propagating the verification of a compound object, from one extremity to another. The subsuming verified supercycles are taken to be the compound objects in the scene (subsumption is by area).

The simpler objects found earlier and these compound objects are further filtered by subsumption. The remaining objects are then grouped into larger objects by continuity of the outer ribbons. The grouped objects may have missing parts (possibly due to occlusion) and these missing parts are filled in by interpolation to form merged objects or superobjects. In subsequent sections we describe each step in greater detail.

4.1 FINDING TERMINATORS

Following are the cases where a ribbon is terminated by a boundary fragment or by a part of itself. The first case is the most important, the others are really exceptions. **(a) Terminator boundary tracing:** This is the most important case. An extremity of a ribbon is verified if edges can be found such that we can traverse from one side of the ribbon extremity to the other. We traverse from one side to the other using a backtracking algorithm implemented as a depth first search. We check if the terminator satisfies the "*in-betweenness*" property, which basically says that the terminator lies completely within the axial contour generators. (Here the axial contour generators are the sides of the ribbon.) The terminator should also satisfy the *terminators-are-extremities* property. That is, it should lie outside the extremal slices of the ribbons (within a tolerance). (Terminators are at the extremities of a generalized cone and are not expected to go deep inside it. Little dips may be allowed due to undulations in the terminators.) If we find several possible terminator boundaries, we pick the best one (shortest, smoothest and with the least number of gaps). The other cases of finding terminators are: **(b) closed curve axis, (c) sharp taper of ribbon** and **(d) perpendicularity of extremities-of-sides to axis.**

4.2 FINDING JOINTS

After finding terminators for ribbons, we pick out those ribbons that have terminators at both their extremities. These ribbons are completely verified and are declared objects. (They are simple objects.) They will not participate in joint formation and are removed from consideration in the subsequent steps in this subsection. Figure 8 gives the block diagram of the process for finding compound objects.

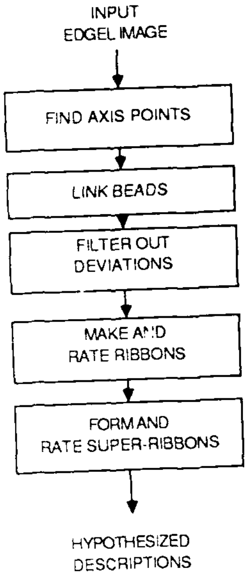


Figure 5: Block diagram of the hypotheses generation part of SHAPE II

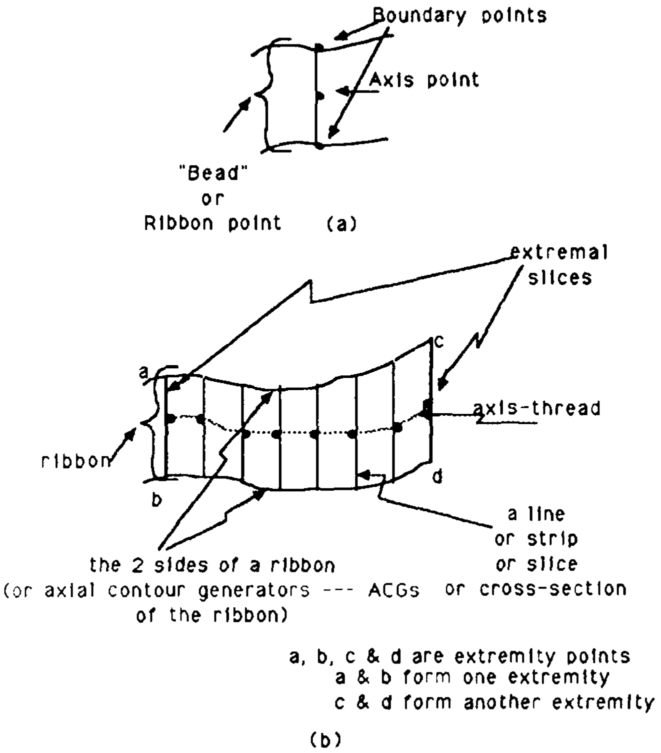


FIGURE 6: THE TERMINOLOGY USED IN AXIS FINDING

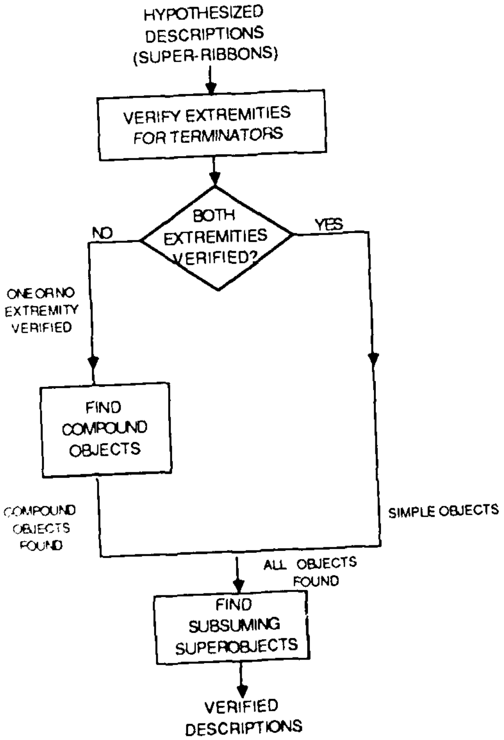


Figure 7 Block diagram of the verification part of SHAPE II

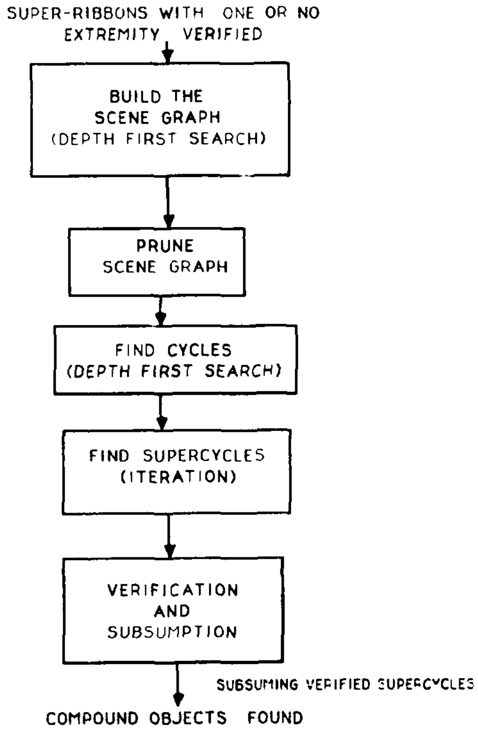


FIGURE 8 BLOCK DIAGRAM FOR FINDING COMPOUND OBJECTS IN THE VERIFICATION STAGE

Representing a scene as a graph: To find joints we represent a scene as a graph called a *scene graph.* Our objective is to represent the scene so that the joints of objects in the scene are cycles in the graph. To do this, we make each extremity point of each side of a ribbon, a node in the scene graph. Thus each extremity is represented by two nodes and each ribbon by two pairs of nodes. (This pair of nodes for a ribbon may be called a super-node, and may be looked at as an entity in its own right.) We form nodes for all the ribbons found in the scene. An *eligible node* is a node whose corresponding ribbon extremity has not been verified by the above terminator finding process.

Building the scene graph: Our next objective is to form links between the nodes in the scene graph. There are three types of links, classified by the way we can traverse from one node to another:

Type1 links between the nodes at the same extremity of the *same* ribbon,

Type2 links between nodes at opposite extremities of the *same* ribbon but the same side of the ribbon, and

Type3 links between the nodes of *different* ribbons.

See Figure 9 for some simple ribbons and their corresponding nodes. We also illustrate the **type1** and **type2** links there.

The last kind of link (**type3** link) is the difficult one to obtain for a scene. For verification purposes we need to form these links among only the eligible nodes (defined earlier). The links from each node are obtained by traversing from it to all other eligible nodes within a certain distance from it. This traversal is like the linking problem for finding terminator boundaries, except that we are now dealing with the case of several possible goal nodes (the other eligible nodes in the scene). We do this again by backtracking using a similar depth first search algorithm. This is done to negotiate breaks in the boundaries and stray surface marks.

As we traverse along a boundary, we do *not* establish links with all the ribbons that we find on our path. Only some of these ribbons are acceptable. The acceptability criteria are based on the following observations (both explained in section 2): **Same-sidedness Observation O2** and **Non-overlapping Observation O3**.

To explain the **Non-overlapping Observation O3**, we first define the term *node-inbetween boundary* between two nodes as the list of edges joining the ribbon extremities corresponding to those nodes. The non-overlapping observation has three parts:

1. *The node-inbetween boundary between the two nodes of two ribbons should not share any edges with the boundaries of the ribbons themselves, except at the extremities of the ribbons.*

2. *The node-inbetween boundary should* not *lie in the area of the two ribbons under consideration (except the ribbon extremities).*

3. *The two ribbons should* not *share any area. (They may share boundary edges.)*

Figure 10 explains this observation.

We now explain the formation of the remainder of the graph using **type3** links. Consider a node, called *current node*, representing a ribbon called *current ribbon.* Consider another node called *other node*, representing another ribbon, called *other ribbon.* For other node to form a link with the current node, the other ribbon and the current ribbon should satisfy the constraints set by the above observations. Figures 11 shows a complicated schematic example with shadows, markings and breaks in the boundaries; for simplicity we consider only some of the ribbons in the image. Some of the ribbons shown are parts of the object and some are due to shadows and markings. The problem is to retain those ribbons that are good candidates for parts of the object and to filter out those due to shadows and marks. We first form the graph with these ribbons and this is shown in Figure 12. Some arcs are shown light with cross-marks and a note explaining why they are *not* valid.

Pruning the scene graph: We can prune the scene graph so that the search space becomes smaller when we look for cycles. We prune the search space by keeping only those nodes corresponding to ribbons that have at each extremity: either a terminator or connections to other ribbons on each side of that extremity. The other nodes are pruned and links involving them are removed from the graph. Figure 13 shows the pruned graph. Ribbons R7, R8, R9, R10, R11 and R12 have extremities without terminators and without connections with other ribbons. They are pruned out and arcs connecting them with the remaining ribbons are removed.

Finding cycles: Our objective is to find cycles in the scene graph because they correspond to joints in the physical objects.

Depth first traversal: We form cycles by depth first traversal of the pruned graph. For each node we check if we can traverse back to that node through a path in the graph. For this we require a method to expand from a node and to obtain children nodes. This is obtained from the **type1** and **type3** links in the graph (and *not* the **type2** links). Note, this process of finding cycles leads to repeated cycles (if the cycle has n nodes, we get n similar cycles). So we modify the process by not starting with a node that is part of a cycle already found. By forming cycles we

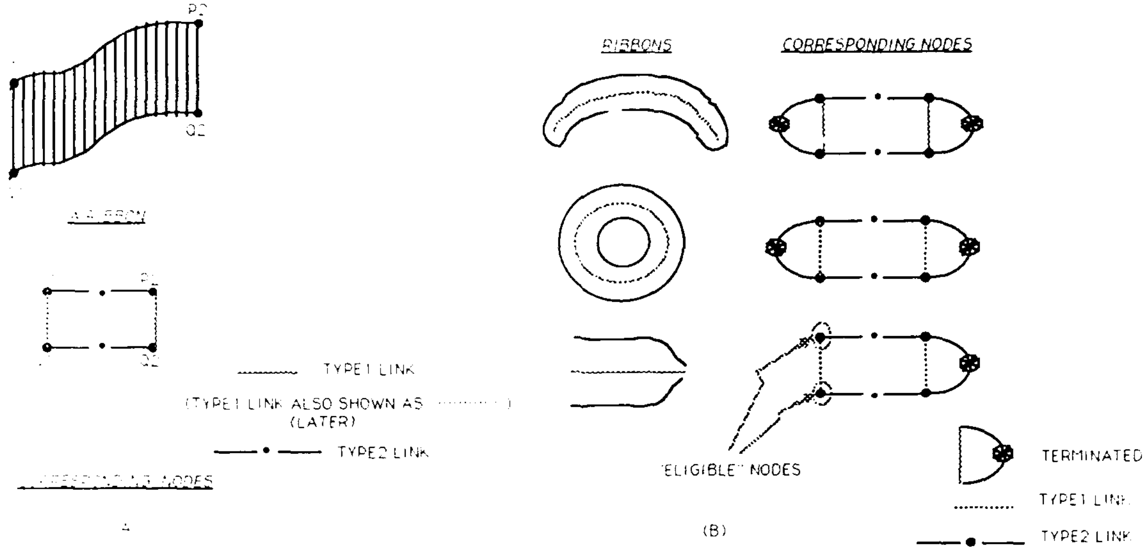
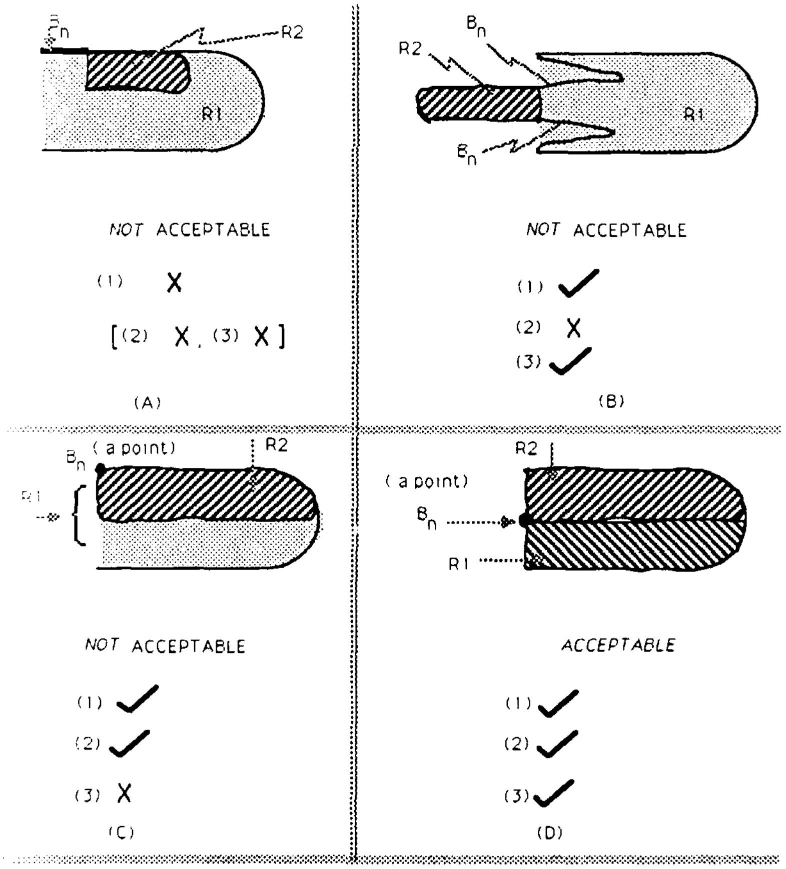


FIGURE 9 THE NODES FOR A RIBBON IN THE SCENE GRAPH



LEGEND

B_n node-inbetween boundary
R1, R2 two ribbons
(1), (2), (3) connote to sub-observations of non-overlapping observation

Figure 10: The non-overlapping observation

have found joints in the objects. Forming joints may be considered a first level in the grouping of ribbons. Figure 14 shows the cycles found for the example scene and the ribbons corresponding to those cycles.

Forming supercycles: A supercycle is a combination of cycles. A supercycle in the scene graph corresponds to a group of joints in the physical object. (A compound object may be looked at as a group of joints with the outer ribbons in the group verified at the outer extremities. The terms used here are defined precisely later.) We thus form supercycles to take the above process of grouping ribbons one step further. The supercycle formation process may be understood as a mechanism to propagate the verification check of an object from one extremity to another. Supercycles are formed by combining cycles in the graph using the **type2** links. That is, we combine two cycles that have a common ribbon (the opposite extremities of the ribbon are in the two cycles). The ribbon is a link between the two cycles. The control strategy we use to combine cycles is iteration.

Iteration. The algorithm for forming supercycles is as follows:

1. **Initial condition:** Initially all the cycles formed are supercycles.
2. **Merging:**
   (a) For each of these supercycles check if it can be merged with any other supercycle.
   (b) If so form a new supercycle.
   (c) Keep a list of new supercycles and the old ones that have not been merged.
   (d) Do this merging in each iteration and keep a marker indicating whether new supercycles have been formed.
   (e) **Stop** the process when there is no change between successive iterations.

This algorithm returns a list of supercycles found.

For each supercycle returned, we check if the extremities of its outer ribbons (ribbons away from the joints, defined more precisely below) are verified with a terminator. If so the supercycle is verified. We throw away the supercycles not verified and keep those that are verified. Of the verified supercycles we take only those whose areas subsume those of others; these are the subsuming verified supercycles. The subsumed verified supercycles may be understood as *substructures* of the subsuming verified supercycles. It may thus be noted that our system is capable of descriptions at different levels (substructure and superstructure descriptions) of a scene.

Figure 15 explains the process of supercycle formation. It shows a supercycle found in the scene graph of Figure 11 using **type2** links to merge the cycles in Figure 14. For convenience of explanation, we introduce some general terminology for supercycles. An *inner ribbon* of a supercycle is a ribbon that has all its nodes connected to other ribbons in the supercycle and none of whose extremities is verified by a terminator. In this example, R2 is an inner ribbon. Call the ribbons away from the inner ribbon, the *outer ribbons* and call their extremities away from the inner ribbon, the *outer extremities.* Here ribbons R1, R3, R6, R4 and R5 are the outer ribbons. (The outer extremities are taken as the "end" extremities for convenience of illustration in this example.) These outer extremities need to be verified for the supercycle to be verified. They are indeed verified for this supercycle.

4.3 FINDING OBJECTS AND MERGING OBJECTS

Objects are the subsuming verified supercycles (compound objects) and those ribbons that were verified earlier by terminators at both their extremities (simple objects). From these we filter out the objects subsumed by others in area. (Note that we again have substructure descriptions available.) We then group the subsuming objects by continuity of the axes and the widths of the outer ribbons. (The outer ribbon of a simple object is the single ribbon itself, describing the object.) The grouping of objects is done by an iteration process very similar to that of finding supercycles. The grouped objects may have missing parts (possibly due to occlusion) and these missing parts are filled in by interpolation of the ribbons hypothesized to be joined. The merged objects or *superobjects* are the final object-level descriptions of the scene. Figure 16 gives the block diagram of this stage. The method is explained by parts (d), (e) and (f) of Figure 18, the scene of two hammers with occlusion.

5 RESULTS AND CONCLUSION

The system described above has been implemented in Common Lisp on a Symbolics 3600 series machine under Genera 7.2. It has been tested on a number of scenes with good results. We show some of the examples here, more results can be found in [10]. For each case we show the input edge data and the final descriptions obtained by our system. All the examples show compound objects. In all the examples we have breaks in the boundaries and marking on the object and background. (In some cases the marks on the background are shadows.) In these examples the breaks are non-trivial, because mere linking of the edges does not generate closed boundaries — due to the markings being closer to the boundary fragments. The first example (Figure 17, synthetic data) shows results on a plane (a compound object). Here we have supercycles being formed by combining two cycles. Figure 18 (synthetic data) demonstrates the capability of our system in describing compound objects with occlusion. Figures 19 show results on a complex real image with occlusion and considerable texture in the background and on the objects. We show the Canny edges [4] for these images. A large number of hypotheses are generated for these examples and the ribbons verified among them are displayed. These examples demonstrate that our system can obtain good descriptions from complex real scenes with occluding objects and a lot of texture.

To give an idea of the detailed running of the system we give in Table 1 specific numbers at various stages of the program on the two hammers example (Figure 18). The image has 181 columns and 111 rows; it has 1034 edgels. 7240 axis points are found, which give 872 ribbons and then 261 super-ribbons. These super-ribbons form 1044 possible nodes in the graph, which is pruned to 32 nodes corresponding to ribbons that have a potential to be verified. From these nodes 4 cycles are found; these are also the supercycles found (as there is no merging of several cycles in this example, unlike in the plane and ship cases). All these supercycles are verified at their extremities. From these we get only 2 supercycles that subsume the others. These are the subsuming verified supercycles and are the compound objects in the scene. In this example, one simple object is found; the total number of objects found (simple and compound) is 3. These are also the subsuming objects found. Two of these objects are grouped together and merged to form one superobject. The number of superobjects found is 2.

We hope that these examples suffice to show the power of our approach for computing structured, symbolic descriptions from realistic input which consists of fragmented boundaries including those caused by shadows, markings and noise. We do not claim that our system will describe *every* scene presented to it in the same way that humans will. The scenes we can handle are those that consist of objects well described by GCs. We would also like to reiterate that *a strength of our system is that our results are very stable with respect to the weights for ratings of ribbons and the preference heuristics based on ribbon attributes.*

The descriptions generated by our systems can be used for a variety of purposes. They should suffice for recognition in most cases. It has been observed elsewhere (see, for example [8]) that the axes of objects are often sufficient for their recognition. We compute only the 2-D projections of 3-D axes, but their connectivity structure may be distinct enough for recognition in many cases. Of course, one can find instances where the full 3-D description is needed. In such cases, we believe that our descriptions provide a good intermediate step. We believe that symmetry descriptions play a key role in inferring 3-D shape from 2-D. We also believe that good monocular descriptions greatly aid in the process of stereo analysis and that the descriptions generated by our system are of the right level.

References

[1] H. Blum. Biological shape and visual science (part 1). *Journal of Theoretical Biology*, 38:205–287, 1973.

[2] J. M. Brady and H. Asada. Smoothed local symmetries and their implementation. *The International Journal of Robotics Research*, 3(3):36–61, 1984. Fall.

[3] R.A. Brooks. *Symbolic Reasoning among 3-D Models and 2-D images*. Technical Report AIM-343, Stanford Artificial Intelligence Laboratory, June 1981.

[4] J. Canny. A computational approach to edge detection. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, PAMI-8(6):679–698, November 1986.

[5] H. Lim and T. O. Binford. Structural correspondence in stereo vision. In *Proceedings of the DARPA Image Understanding Workshop*, pages 794–808, April 1988.

[6] D. Marr. Analysis of occluding contour. In *Proceedings of the Royal Society of London*, pages 441–475, 1977.

[7] R. Mohan and R. Nevatia. Perceptual grouping with applications to 3-D shape extraction. In *Proceedings of the IEEE Computer Society Workshop on Computer Vision*, pages 158–163, November 1987.

[8] R. Nevatia. *Machine Perception*. Prentice Hall, 1982.

[9] R. Nevatia and T.O. Binford. Description and recognition of complex-curved objects. *Artificial Intelligence*, 8:77–98, 1977.

[10] K. Rao. *Shape description from sparse and imperfect data*. PhD thesis, University of Southern California, December 1988. Also available as technical report USC-IRIS-250.

[11] K. Rao and R. Nevatia. Computing volume descriptions from sparse 3-D data. *International Journal of Computer Vision*, 2(1):33–50, 1988. Also as book chapter in Advances in Spatial Reasoning, Ablex Publishing Corporation, 1989.

[12] S. A. Shafer. *Shadow Geometry and Occluding Contours of Generalized Cylinders*. Technical Report CS-83-131, Carnegie-Mellon University, May 1983.

[13] F. Ulupinar and R. Nevatia. Using symmetries for analysis of shape from contour. In *International Conference on Computer Vision*, December 1988.

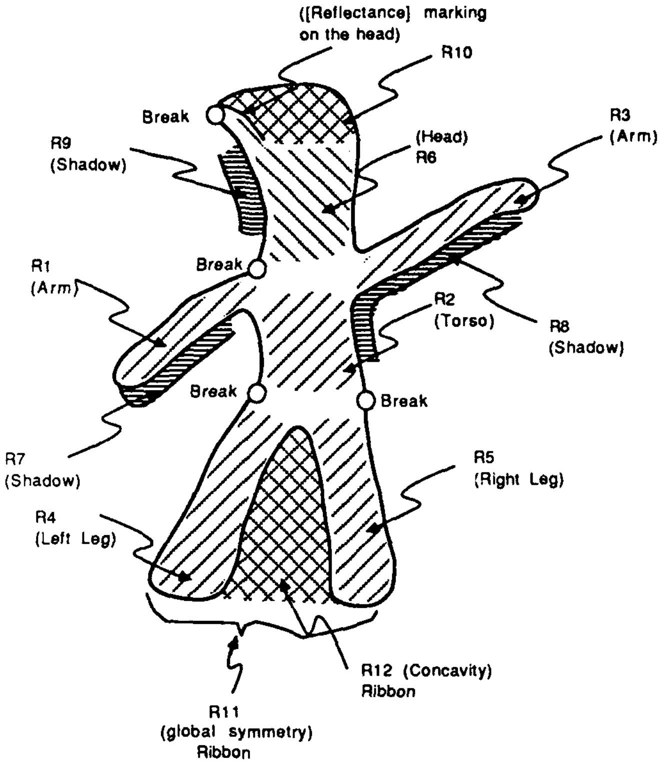


Figure 11: An example of a *compound object* (Gumby) with shadows, marking and breaks.

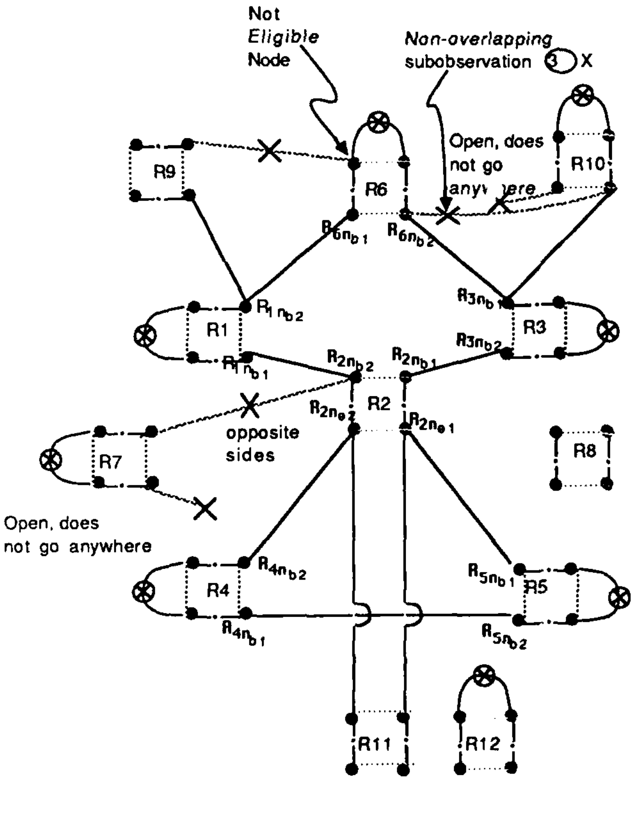


FIGURE 12: THE GRAPH FOR THE GUMBY EXAMPLE. SOME ARCS ARE SHOWN LIGHT WITH CROSS-MARKS AND A NOTE EXPLAINING WHY THOSE ARCS ARE NOT VALID.

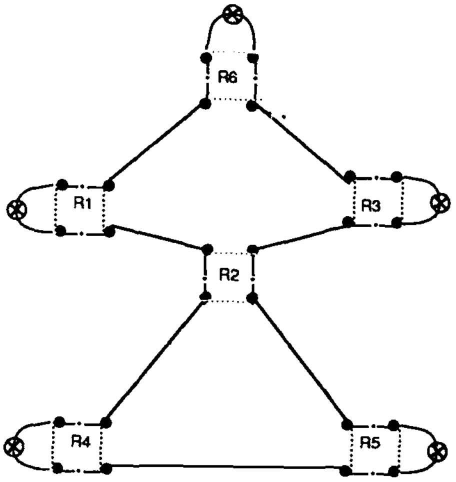
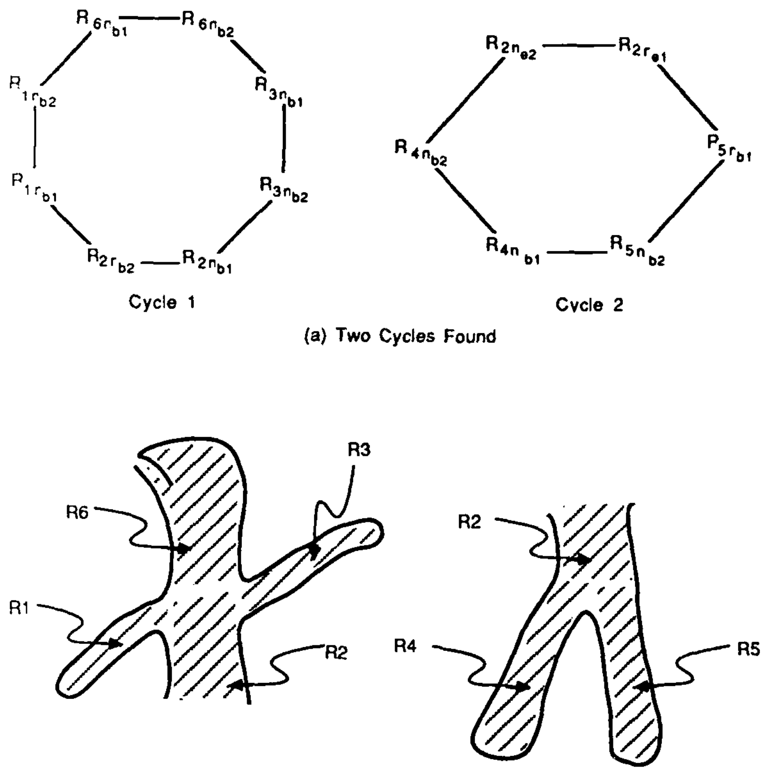


FIGURE 13: THE PRUNED GRAPH FOR THE GUMBY EXAMPLE



Cycle 1

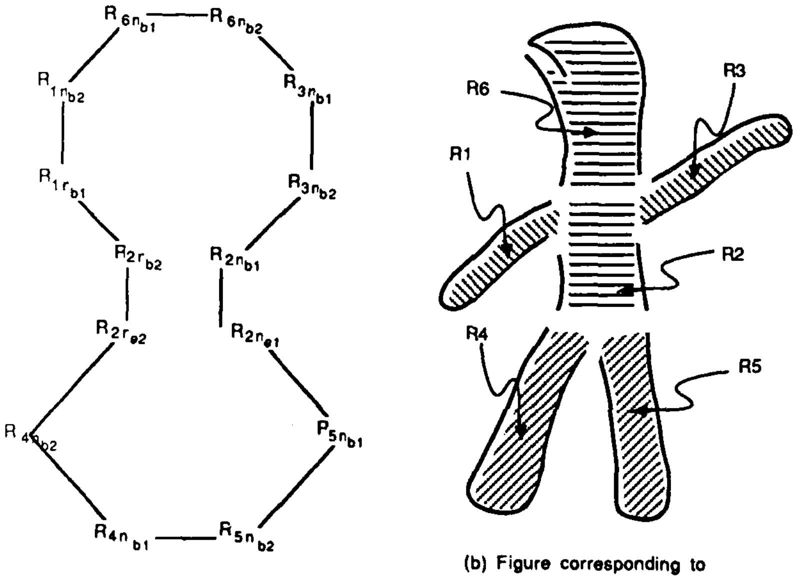
Cvcle 2

(a) Two Cycles Found

(b)(i) Figure corresponding to Cycle 1

(b)(ii) Figure corresponding to Cycle 2

Figure 14: Cycles found for the Gumby example. Also shown are the ribbons corresponding to the cycles.



(b) Figure corresponding to the supercycle found. This is also the only subsuming verified sypercycle and the only superobject

(a) Supercycle

Figure 15: (a) The supercycle found for the Gumby example. (b) The ribbons corresponding to the supercycle.

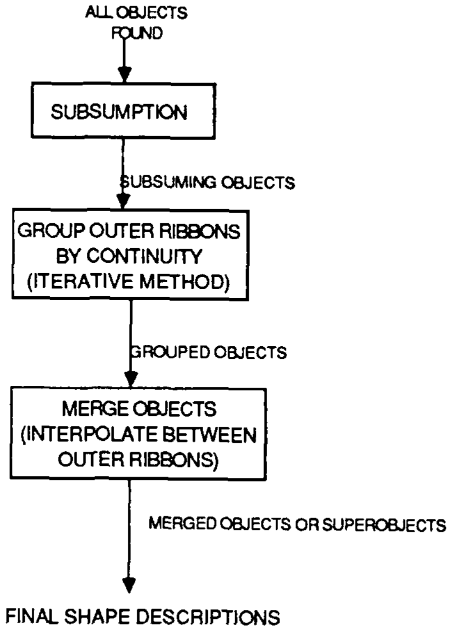

ALL OBJECTS FOUND
SUBSUMPTION
SUBSUMING OBJECTS
GROUP OUTER RIBBONS BY CONTINUITY (ITERATIVE METHOD)
GROUPED OBJECTS
MERGE OBJECTS (INTERPOLATE BETWEEN OUTER RIBBONS)
MERGED OBJECTS OR SUPEROBJECTS
FINAL SHAPE DESCRIPTIONS

FIGURE 16: BLOCK DIAGRAM FOR FINDING SUBSUMING SUPEROBJECTS IN THE VERIFICATION STAGE

Image name	two hammers
Figure number	Fig. 18
Image dimension	181 by 111
# of pixels	20091
# of edges	1034
# of axis points	7240
# of linked lists of axis points	872
# of super-ribbons	261
# of nodes in the graph (all super-ribbons)	1044
# of nodes after pruning	32
# of ribbons after pruning	8
# of cycles	4
# of super-cycles	4
# of verified super-cycles	4
# of subsuming verified super-cycles (compound objects)	2
# of simple objects	1
# of objects (simple and compound)	3
# of subsuming objects	3
# of superobjects	2

Table 1: the values of some variables at various stages of processing for the two hammers example

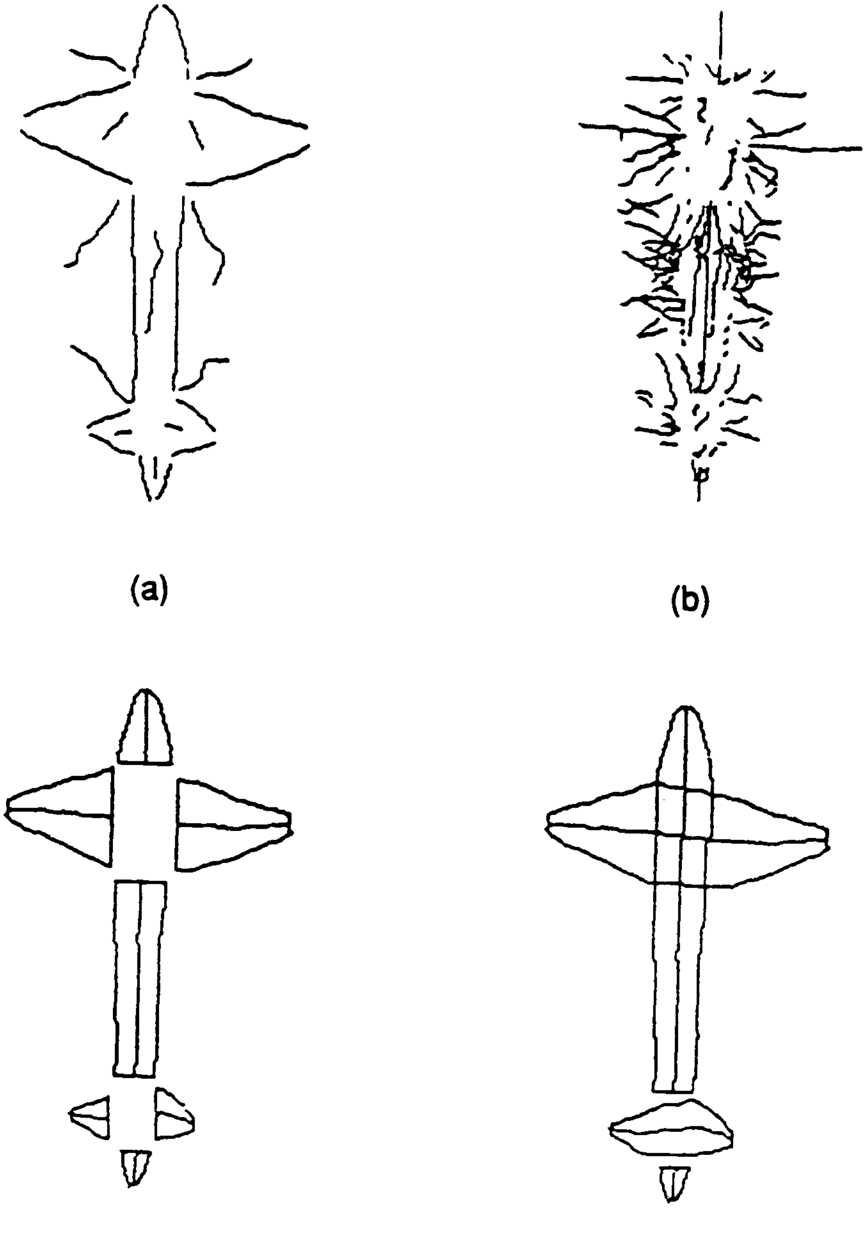


Figure 17: (a) input edge data for plane, (b) all hypotheses generated, (c) supercycle found (grouping of two cycles), (d) final result (superobject)

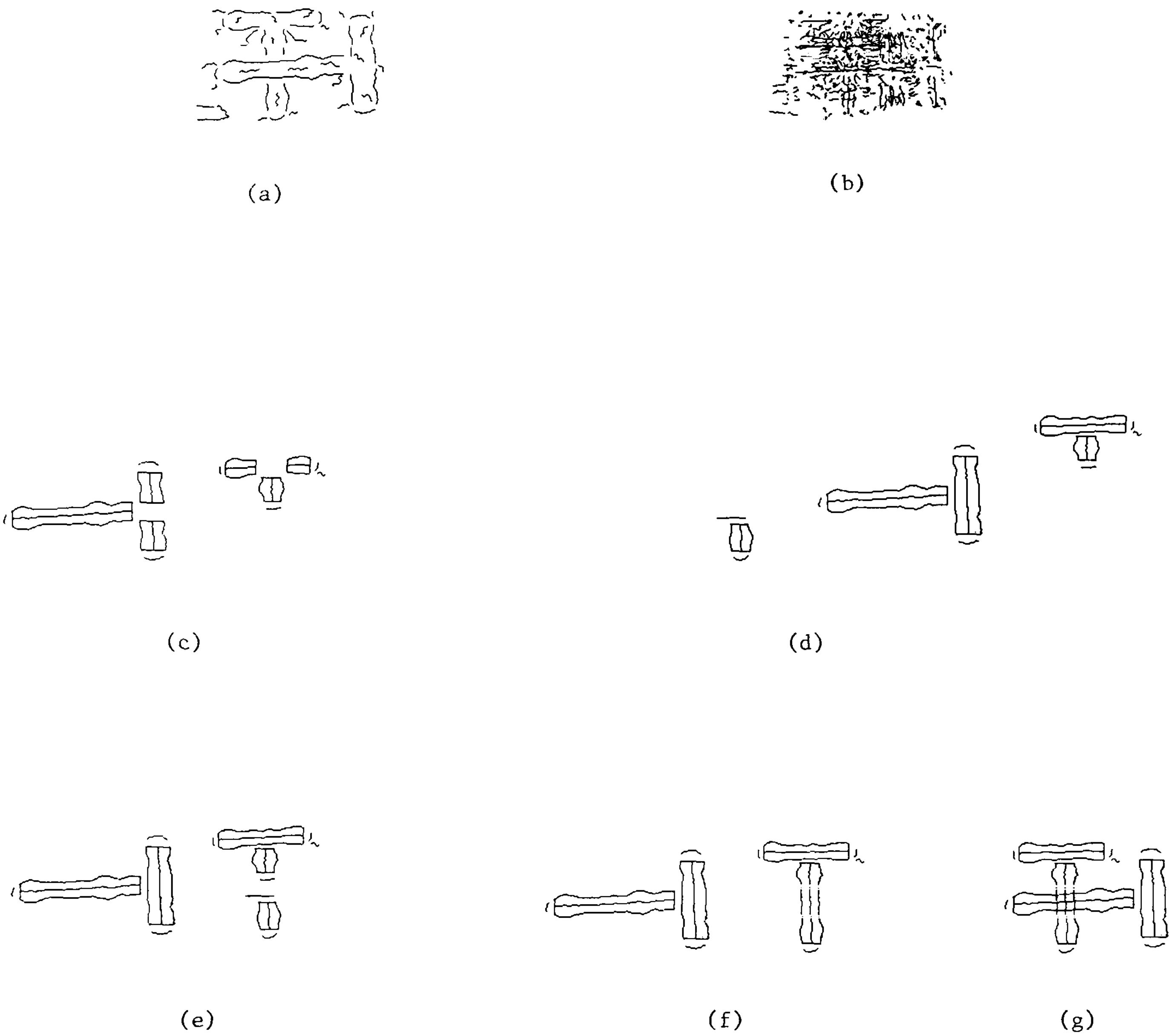
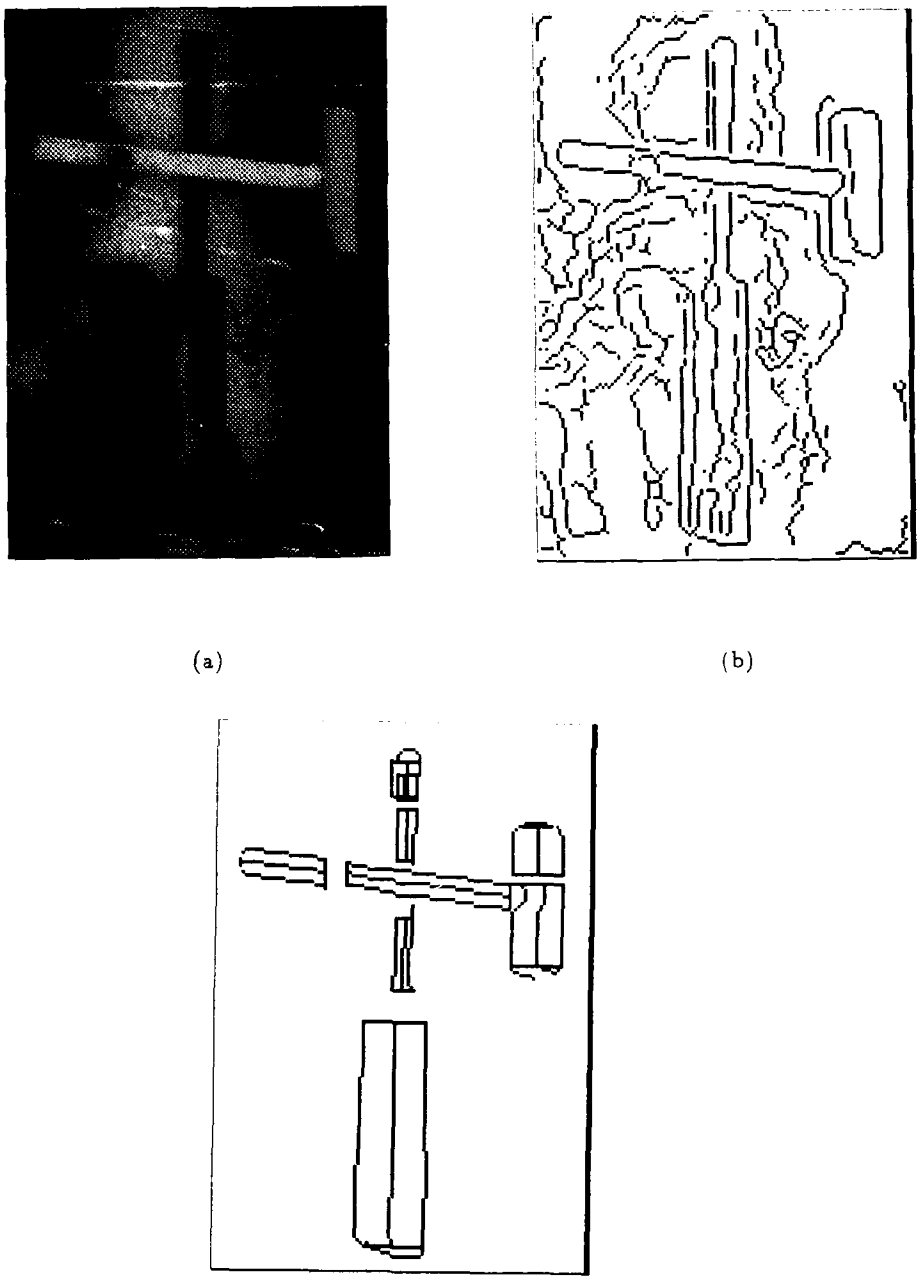


Figure 18: (a) input edge data for two hammers with occlusion, (b) all hypotheses generated, (c) supercycles found (just one cycle each), (d) all objects found, (e) grouped objects, (f) superobjects, (g) final result (superposition of superobjects found)



(a)

(b)

(c)

Figure 19: (a) original image with a lot of texture of a screw driver and hammer, (b) input edge data, (c) final result

PERCEPTUAL ORGANIZATION FOR SEGMENTATION AND DESCRIPTION*

Rakesh Mohan and Ramakant Nevatia
Institute for Robotics and Intelligent Systems
University of Southern California
Los Angeles, California 90089-0273

ABSTRACT

Visual perception involves a *representational framework* for the visual percept and visual processes that operate on the descriptions in this framework. We present a description framework, motivated by *perceptual organization*, which consists of representations of the geometrical organizations of intensity discontinuities. The descriptors in this framework are called *collated features*, and are groupings identified by perceptual organization. We describe the processes that operate on the image to obtain these descriptors, and the visual processes that utilize them. The detection of collated features is robust to local problems. The structural information encoded in them aids various visual tasks such as object segmentation, correspondence processes (stereo, motion and model matching) and shape inferences.

We identify two primary grouping processes, *co-curvilinearity* and *symmetry*, which are applied to intensity edge-contours to generate the collated features, including *curves, symmetries* and *ribbons*. We show that these collations can be used to segment scenes into visible surfaces of objects and to describe the 2-D shapes of those surfaces. We also propose various applications for these collated features.

1 INTRODUCTION

The task of visual perception involves evolving stages of description of the percept. As a vision system processes data, it generates better and richer descriptions which are used for making various interpretations about the the scene. The problem we address here is one of extracting suitable visual descriptors from a scene. We believe that the descriptors should meet the following requirements:

Structure: Since structure plays a crucial role in vision, the descriptors obtained from a scene should describe the structural properties and relationships in it.

Visual Invariance: Features should be *visual invariants*, i.e. descriptors of the visual data that do not change with changes in the parameters of observation, such as illumination and viewpoint.

Physical Significance: Features should correspond to some important physical characteristics of the objects being viewed.

Description: The visual descriptors should be useful in helping the visual system generate a spatial description of the environment.

Let us examine the commonly used features, namely *regions* and *edge contours*, in the context of these requirements. Regions are based on intensity (or color) distributions, edge-contours on local edgel contiguity, thus neither address the issue of describing the structural properties of a scene. Their detection processes are local and are sensitive to local variations. Intensity similarity and edgel contiguity have low physical significance, in that they do not correspond directly to structural properties of the objects being viewed; they are based on intensity rather than structure. Most importantly, region segmentation and contour tracing do not provide any shape description of the entity segmented, making them useless for purposes of reasoning about object shapes or formation of their shape descriptions.

In this paper we present as an alternative, a new visual representational framework, motivated by perceptual organization, which better meets the above requirements. The descriptors in this framework are *curves, points, contours,*

*This research was supported by the Defense Advanced Research Projects Agency under contract number F 33615-87-C-1436 monitored by the Air Force Wright Aeronautical Laboratories, DARPA order no. 3119

symmetries and *ribbons* (section 2). These features are obtained in the following way. First edges are detected and linked into contours. Co-curvilinearity is used to join contour-segments into curves (section 3). Possible symmetry relationships between these curves are hypothesized and the best symmetries selected using a neural-network (section 4). The area bounded by each pair of symmetric curves is a ribbon. To show an application for these descriptors, we use them to segment scenes into the visible surfaces of objects. A 2-D shape description of the surfaces, in terms of ribbons, is automatically obtained (section 5). We propose some other applications for these representations (section 6) and finally present our conclusions (section 7).

2 PERCEPTUAL ORGANIZATION

It has been shown that our visual system can immediately detect such feature relationships as collinearity, parallelism, connectivity and repetitive patterns among image elements [1]. This phenomenon is called *perceptual organization.* We propose that perceptual organization takes primitive image elements typically generated by low-level segmentation processes, and generates representations of feature groupings which encode the structural interrelationships between the component elements. We term these representations *collated features.*

2.1 NON-ACCIDENTALNESS

The principle of *non-accidentalness* [2, 1] states that regular geometric relationships are so unlikely to arise by accident that when detected, they almost certainly reflect some underlying causal relationship. If we detect viewpoint-invariant structural relationships that are common in the objects of interest, then using the non-accidentalness principle we can reason that the detected geometric organizations were caused by the structure of the objects in the scene. We will, therefore, design collated features such that they identify those structural relationships that are most common among objects of our visual domain and remain invariant in 2D projections over most viewpoints, i.e. collated features that identify structural arrangements of image tokens that have a high probability of corresponding to object structures.

The work on collated features, in computer vision, has been limited. The authors have used collated features in a system to detect buildings in aerial images [3]. In this system, the collated features are used to perform stereo matching and to generate 3D models and shape descriptions of the buildings. Simple collations have also been used to limit the computational complexity in image to model matching [1, 4].

2.2 THE DESCRIPTION HIERARCHY

While a number of geometrical relationships have been proposed as detected by perceptual organization, we believe that two primary relationships, that of *co-curvilinearity* and *symmetry,* when applied repeatedly to image tokens (or their simple representations such as end-points or axes) can account for most of the groupings. Some of the geometric relationships are particular cases of these two primary relationships; for example, collinearity is a special case of co-curvilinearity and parallelism of symmetry. Other geometrical relationships are encompassed by them, for example proximity and connectivity are parts of co-curvilinearity[1].

The collated features generated by application of these two geometric organizations on intensity edges are:

Curves: A smooth curve through contiguous edgels (with possibly some gaps) with no tangent discontinuities or extremas of curvature.

Points: Terminations and junctions of curves.

Contours: A contour is an ordered set of contiguous curves.

Symmetries: Pairs of mutually symmetric curves.

Ribbons: A ribbon is the area enclosed by two symmetric curves, and is described by a *symmetry axis* and a *sweeping rule.* The sweeping rule gives, for each point on the axis, the corresponding pair of symmetric points on the two curves.

The geometrical relationships chosen are viewpoint-invariant. Co-terminations (junctions) and co-curvilinearity among 3D curves project with the same relationship in 2D [5, 1]. While we will also consider curvature-extremas

[1] In this paper we are interested in developing structural descriptors from edges. We will, therefore, ignore geometrical organizations, such as repetitive patterns which are more suitable for handling textures. We will also, for simplicity, ignore effects of intensity and color on grouping.

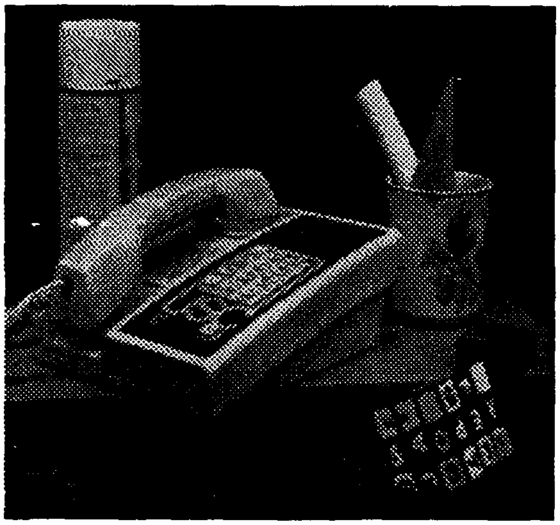


Figure 1: Image I

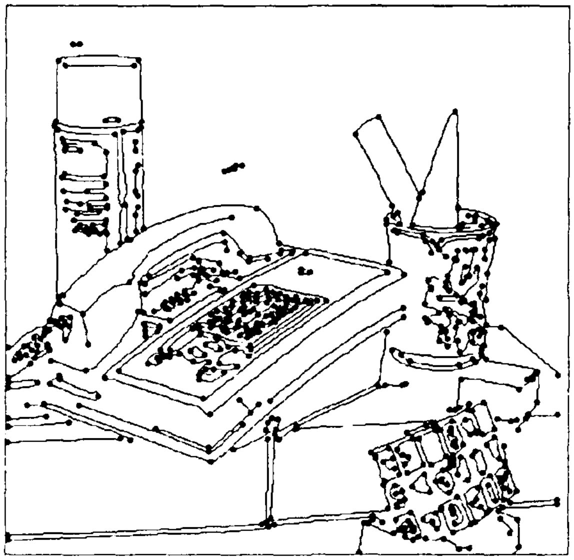


Figure 2: Edge contours and corners

The geometrical relationships chosen are viewpoint-invariant. Co-terminations (junctions) and co-curvilinearity among 3D curves project with the same relationship in 2D [5, 1]. While we will also consider curvature-extremas as junctions or corners, they may change specially in foreshortened views. However, people have been shown to be sensitive to curvature-extremas [6]. The symmetry relationship we detect is also largely viewpoint-invariant, and is discussed in more detail in section 4. Surface boundaries of most objects are composed of smooth curves (co-curvilinearity), and animal shapes and man-made objects exhibit symmetry.

3 CO-CURVILINEARITY

Indvidual curves are grouped into curvilinear structures on the relationship of *co-curvilinearity*. Co-curvilinearity of curves can be broken into two components: *continuity* and *proximity* of similarly oriented curves.

3.1 CONTINUITY

Collations of continuous curves are detected by a local, non-iterative process that selects the most collinear, or least bent, joins among neighboring curves. Our formulation of the continuity based organization is influenced by experiments [7] that indicate that the grouping process (at least for point like tokens) can be modeled by local selection of most collinear pairings.

Edges are detected using a Canny edge-detector [8] and are linked into contours using a simple algorithm based on eight-neighbor connectivity. See figure 2 for edge contours obtained for an image. The edge-contours contain mistakes caused by wrong linking at forks (which can not be seen by a visual inspection of figure 2), linking errors caused by edge displacement at junctions (for example, the junction of the wedge and the cylinder inside the cup in figure 2), and edge dropouts due to poor contrast.

The edge-contours so obtained are segmented into curves at curvature extremas. We use adaptive smoothing [9] to detect curvature extremas (marked in figure 2 by points). We then apply the continuity based grouping process on these curves. The neighborhood searched for possible joining curves is constrained to a small area at the end of each curve (about 10 pixels) and the smooth joins are made as outlined above. The gaps are filled by approximating the curve in the gap by a straight line.

3.2 PROXIMITY

We also group overlapping, proximate, similarly oriented curves into curvilinear structures. This grouping is motivated by issues of scale; proximate parallel curves are interpreted as boundaries of surfaces (or ribbons) rather than as representing multiple, very thin ribbons, which would have indicated multiple, narrow surfaces. Also, in performing this grouping, we avoid the problem of multiple symmetries being detected between two groups of parallel, proximate curves, for a single surface bounded by the two curve-groups.

the symmetry axis between them (the complete axis is not required, it is sufficient to have a few points on the axis), and check that the orthogonal distance from the axis to each curve is roughly the same at various positions along the axis.

Next we form equivalence sets of curves based on the relationship of proximate-parallelism. Each set is a grouping of co-curvilinear curves (on the basis of proximity). For reasons we shall not go in here, the various techniques we have considered, such as least squares fitting of parametric curves, of finding a curvilinear representation of these groups, are too expensive and/or unsuitable. We have found representing a grouping by the longest curve in it, a simple yet effective solution. Proximate co-curvilinear groupings found in Figure 2, and so represented, are shown in figure 3.

4 SYMMETRY

We define the *symmetry* relationship as a one-to-one mapping (if the curves are considered infinitesimally divisible) between the points of two curves with the symmetry axis defined as the locus of the mid-point of the straight lines (or for generality, some curve) joining points on one curve to their image in the other.

4.1 PREVIOUS WORK

A good survey and analysis of symmetry axes[2], specially in the context of ribbons, can be found in [10]. The types of axes defined, and the literature on their properties and their detection is vast; we will not attempt to present an account of them. We will, however, like to point out two important deficiencies in a broad class of these axes.

Firstly, most of the axes have been defined in the context of generalized cones [11, 12, 13], and are suitable for symmetry relationships between limb or extremal boundaries of (straight homogeneous) generalized cones [13, 14, 15, 16, 17]; not for boundaries arising from orientation discontinuities. The line joining symmetric points is restricted to being orthogonal to the symmetry axis, thus cases of skew symmetry [18] are not handled.

Secondly the detection techniques for these axes are based on matching points on the contours, i.e. for each point on a contour, possible matches to all the points on the other contour have to be considered. Thus these detection techniques have a computational complexity of $O(n^2)$ where n is the number of edgels on a contour (this can be reduced to $O(nk)$, k constant, by quantizing the search space, as proposed by Nevatia [14]). Point by point matching means utilizing some measure, such as tangent direction, at each point to evaluate the match. Not only are such measures noisy along real edge-contours, they usually do not vary much along a contour, making the localization of matches difficult. A better scheme would require matches at well localized positions, namely curvature extremas and tangent discontinuities, along the contours.

4.2 TRANSFORMATIONALLY INVARIANT SYMMETRIES

We define the symmetry axis between two curves as the locus of the mid-points of the lines joining points at equal length ratios along the curves. Consider figure 4. Let the length of curve AB be s_1 and that of CD be s_2. A point x on AB is mapped to a point y on CD iff, given that the length of curve section Ax is a and that of Cy is b, $a/s_1 = b/s_2$.

For the detection of this axis between two contours, we need only match the curvature extremas of the contours, as the match for the points between curvature extremas is automatically defined. Thus, detection of this symmetry axis involves matching of curves (recall that curves are sections of contours bounded by corners and terminations) rather than edges.

This axis is also invariant to viewpoint transformations for important classes of curves and for specific symmetry relationships. Consider imaging situations with linear transformations such as orthographic projections and "limited perspective":

$$[\,T\,]\begin{bmatrix} x \\ y \\ z \end{bmatrix} = \begin{bmatrix} \alpha_1 x + \beta_1 y + \gamma_1 z \\ \alpha_2 x + \beta_2 y + \gamma_2 z \end{bmatrix}$$

Mid-points map to mid-points in the projection:

[2] we will use the term "symmetry axes" for the various representational axes employed in vision literature, even when the author(s) may not have termed the representation as that of symmetry.

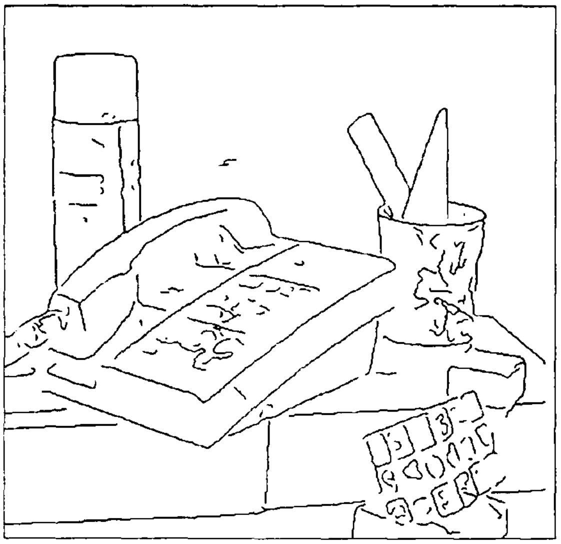


Figure 3: Curves obtained after grouping on proximity

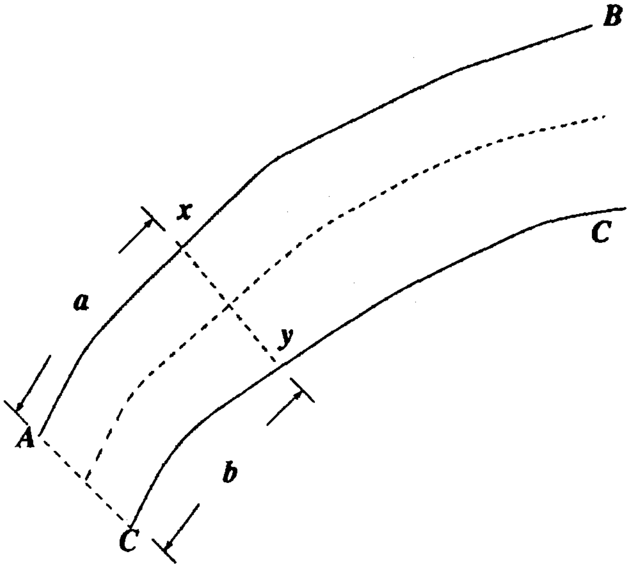


Figure 4: Transformationally Invariant Symmetry Axis (TISA)

$$\begin{bmatrix} \frac{1}{2}((\alpha_1 x_1 + \beta_1 y_1 + \gamma_1 z_1) + (\alpha_1 x_2 + \beta_1 y_2 + \gamma_1 z_2)) \\ \frac{1}{2}((\alpha_2 x_1 + \beta_2 y_1 + \gamma_2 z_1) + (\alpha_2 x_2 + \beta_2 y_2 + \gamma_2 z_2)) \end{bmatrix} =$$

$$\begin{bmatrix} \frac{1}{2}(\alpha_1(x_1 + x_2)) + \frac{1}{2}(\beta_1(y_1 + y_2)) + \frac{1}{2}(\gamma_1(z_1 + z_2)) \\ \frac{1}{2}(\alpha_2(x_1 + x_2)) + \frac{1}{2}(\beta_2(y_1 + y_2)) + \frac{1}{2}(\gamma_2(z_1 + z_2)) \end{bmatrix}$$

Therefore, the mid-points of the lines joining symmetric points are invariant under linear transformations. If in addition, the length ratios of curves are maintained under the transformation, then the axis would b· invariant to these transformations. In general, the length ratios of curves are *not* maintained under all linear transformations. However, for the special case cf straight lines, the length ratios are maintained under these linear transformations (for a proof, replace $\frac{1}{2}$ by any fraction in the above equations). Thus, for straight lines this symmetry axis is invariant to viewpoint transformations.

The axis is also invariant to viewpoint transformations for curves for some restricted, but important, types of symmetries. For example, Ulupinar and Nevatia [19] have proposed two specific symmetries, namely mirror and parallel symmetries, which are useful for determining shapes of surfaces (shape-from-contour). In case two curves have either of these symmetry relationships, then the symmetry axis defined here corresponds to the mirror or parallel symmetry, as the case may be, for all viewpoints.

For each curve in the image (rather for each curvilinear grouping, found as explained in the previous section), we check all other curves in the image for possible symmetrv relationships. To limit the number of symmetries investigated we use these hueristics: First, the shorter of the two curves should be no less than one third the length of the longer curve. Second, there should be certain minimum overlap between the two curves. Curve pairs which meet these two requirements are hypothesized as possibly being symmetric. However, the symmetry axes between them is not calculated (as measurements for evaluating the symmetries can be made by just knowing a few positions along each axis), and the axes are provisionally represented by their simpler straight line counterparts. Figure 5 shows all the symmetry axes hypothesized for the curves in figure 3.

Next we evaluate the symmetry hypoth ·es and select the (few) best symmetries from their alternates.

4.3 EVALUATION AND SELECTION OF SYMMETRIES

Each side of a curve can bound at most one surface. We would, therefore, ideally like to pick at most one symmetry axis for each side of a curve. As any quantitative measure of symmetry is not guarantied to select only those

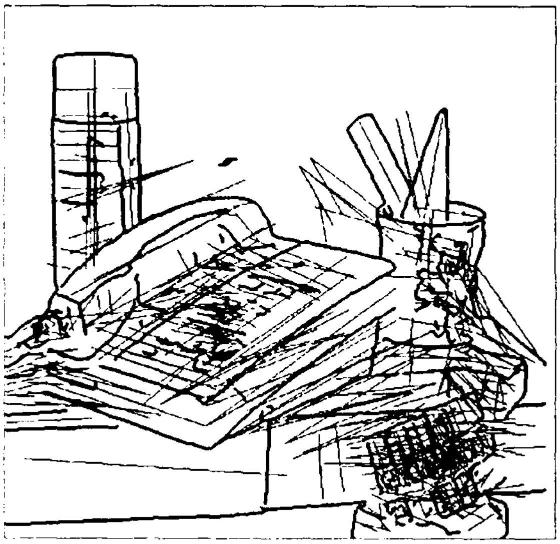


Figure 5: All the symmetry axes hypothesized

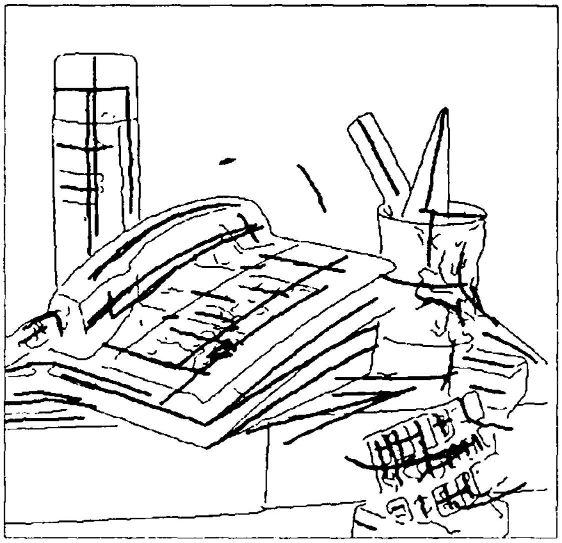


Figure 6: Axes selected

symmetries that are between curves bounding the same surface, it is preferable to allow more than one symmetry axis to be selected for each curve-side.

The selection of the best collations (symmetries) given a set of constraints (namely *conflict* between various symmetries for the same curve-side, and *evaluation* of each symmetry on the basis of geometry) on them, is modeled as a constraint satisfaction problem [3]. We use Hopfield networks [20] to implement a constraint satisfaction network. In this network, each axis is represent by a node (or a neuron). Each of its competitor is connected to it by a negatively weighted link. The value of each axis is comp''ted as a weighted sum of a numerical representation of certain measures on it, and is fed as input to the node corresponding to that axis. This input is computed using the following measures: *cover* (the cover of an axis is the sum of the lengths of the curves it is the axis for), *aspect ratio* (the ratio of the axis length to the distance between the two symmetric curves), the similarity between the length of the two symmetric curves, amount of *skew* between the curves, amount of skew between the ends of the curves (estimated as the amount of skew between straight-sides closing the ends of the ribbon formed by the symmetry relationship), *parallelism* between the two curves, and parallelism between the ends of the curves. The network converges after a few iterations (about 5 iterations, with one iteration representing one time constant and each iteration itself implemented as 10 sub-iterations), and the nodes with high output (> 0.8) are considered selected while the rest are rejected.

In addition to the axes selected by the constraint satisfaction network, we also select those axes where the symmetric curves are joined, at least at one end, by a single curve. This structural relationship is used to ensure that symmetry-axes which have low input (due to low aspect-ratios or high skew) but high chances of corresponding to surfaces (due to simple closure at one end) also get considered for forming ribbons. Figure 6 shows the axes selected from figure 5. Note that the selected axes have been completely computed.

5 SEGMENTATION AND DESCRIPTION

Each selected symmetry axis describes a ribbon, or the area enclosed by the pair of symmetric curves and the edge-contours closing the ends of the ribbon. Our next step is to close the ends of the ribbons with the contours composed of the curves detected in the image. These ribbons are useful for segmenting scenes into visible surfaces. Some surfaces may have complex shapes, and may be segmented into more than one ribbon. For each surface, the component ribbons automatically provide a shape description for the surface.

The search for contours closing the two ends of a ribbon proceeds in the following order:

1. If the two curves share a point then that junction forms the closed end (alternatively, a curve of zero length can be assumed to close that end).
2. The curves in a ꞉D scene can be represented as a graph structure. The curves act as arcs, and the points (curve terminations and junctiυns) act as nodes of the graph. In practice, we represent all curve terminations within

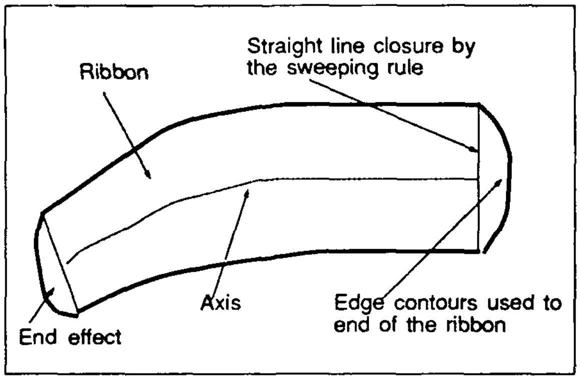


Figure 7: Ribbon

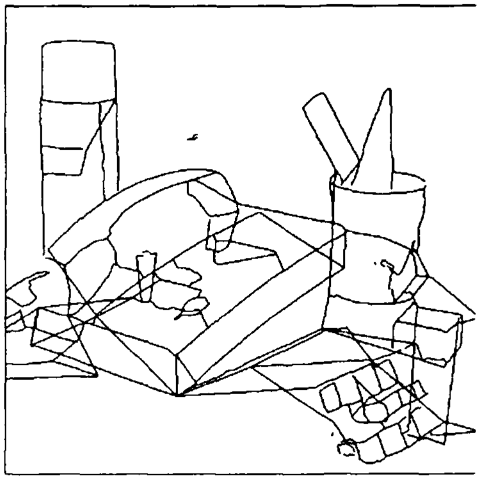


Figure 8: Initial ribbons formed

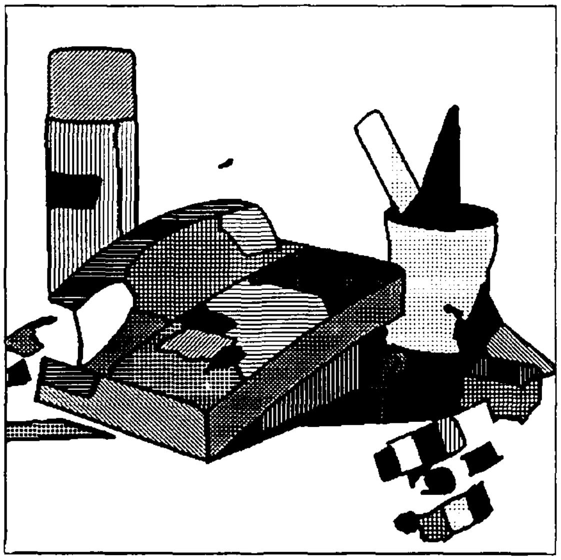


Figure 9: Selected Ribbons

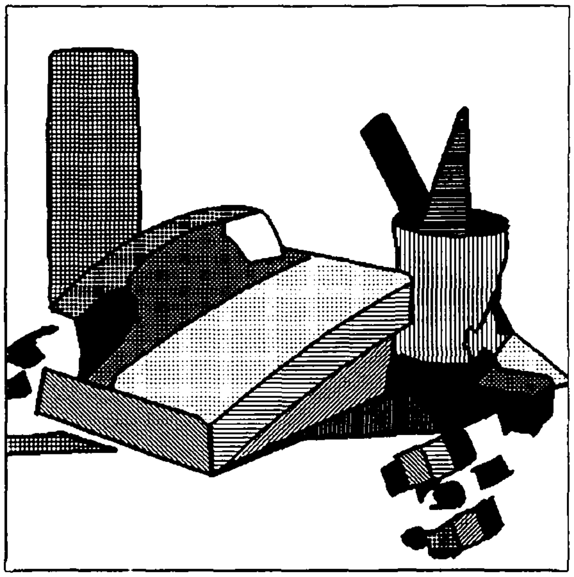


Figure 10: Ribbons after removing those forming false T-junctions

a certain neighborhood by a single node. A best-first search is done on this graph structure, between the two nodes corresponding to the two terminations of the symmetric curves bounding a ribbon, for a path. The path is represented as the contour closing that ribbon end.

3. In cases of poor contrast differences between adjoining surfaces there may be no candidate found in the above search. The next step is to look at curves that lie in the areas between the two points. If some curves are detected in this area, we apply the continuity grouping process on these curves, (as described in section 3) with relaxed constraints on the continuity. The allowable gap is increased to the size of the gap in the ribbon end, and more "bend" is allowed in the joins. The shortest contour so formed is selected.

4. If no contours are found by the above techniques (this may be due to either missing edges or complex shaped surfaces, where a single surface may be represented by more than one ribbon) a straight line join between the two curve ends is proposed as the closure for that ribbon end.

Ribbons so formed are show in figure 8.

5.1 REASONING ON MONOCULAR STRUCTURAL RELATIONSHIPS

In the above process, for those ribbon-ends where no candidate edge-contours were found for closure, we may have proposed straight-lines as closures. If we assume that the scene is composed of opaque surfaces, those ribbons for which the proposed straight-line closures cross over edge boundaries of other ribbons, are potentially wrong. For these ribbons, we try to find an alternate path along the curves (now augmented by proposed joins) which does not cross over existing ribbons. If for a ribbon, no such suitable path is found to replace the wrong straight-line closures, it is rejected. Further, we may reject those ribbons which have a substantial portion of their boundaries represented by proposed straight-lines which neither have edge-support, nor form shared boundaries with another viable ribbon. The ribbons so obtained are shown in figure 9.

Assuming general viewpoint and low probability of accidental alignments, a T-junctions indicates occlusion, with the stem of the T being occluded by a surface whose boundary includes the top of the T (the stem of T obviously does not lie in this occluding surface). Therefore, if we have a ribbon **A** enclosed inside a bigger ribbon **B** with at least one boundary of **A** (not shared by **B**) terminating at a T-junction with **B**'s boundary, then (unless **B** is found to be occluded along the boundary section contributing to the T-junction), this configuration does not support the occlusion interpretation. Thus we can reject ribbon **A** as it forms a *false T-junction* (the ribbon **A** most probably resulted due to surface markings on the surface enclosed by ribbon **B**). Ribbons that remain after removing those that form false T-junctions are displayed in figure 10.

6 APPLICATIONS

As demonstrated in the previous section, collated features are excellent for object segmentation and for generating shape descriptions. Monocular interpretations of shape from contour depend on obtaining both good contours in the image (i.e. connected contours, corresponding to surface boundaries and not texture or markings) and a good axis finding algorithm (for those based on concepts related to skew symmetry [18]). The system presented finds connected contours (ribbon boundaries) which, due to the non-accidentalness principle, have a high probability corresponding to surface-boundaries rather than texture. Also the axes obtained often corresponds to the skew symmetry axes (always for straight lines). In cases of parallel and mirror symmetries [19], the axes detected can be used directly for monocular shape analysis.

The collated features detected, specially curves and ribbons, are invariant to viewpoint-transformations, and are thus excellent tokens for correspondence when images containing objects viewed from different viewpoints have to be matched, for example in stereo and motion matching. We will demonstrate the utility of ribbons for correspondence by using them as match primitives for stereo images which have large vertical disparity.

A ribbon has two axes describing it; the primary axis between the symmetric curves and the secondary axis between the contours closing the ribbon ends. For each ribbon we select the axis with the maximum vertical extent. We define an *augmented epipolar window* for each vertical axis as a rectangle in the other image of the stereo pair with its vertical dimension equal the vertical extent of the axis plus twice 5% (the allowed vertical disparity) of the height of the image, and its horizontal dimension twice 25% (the allowed disparity) of the image width and centered at the center of the axis when projected on the other image. A match is postulated if at least half of the matched ribbon lies in this window, and the corresponding axes (both primary and secondary) of the two matching ribbons do not differ in length by more than 100%, and the difference in orientation of the two axes is no more than 30 degrees.

These three constraints, epipolar overlap, length and orientation similarity, are sufficient to assign unique matches to most of the ribbons in the stereo pair. In case of ambiguous matching (i.e. if some ribbons in the left and/or right image have multiple matches) we pick the best match using a constraint satisfaction network.

7 CONCLUSIONS

We have proposed a representation framework using *collated features* as the representations computed by the process of perceptual organization applied to the primitive image elements. These collations represent structural relationships between the arrangement of their tokens. We have identified the structural relationships so represented, in terms of their *significance* for the shapes in our visual domain and their *utility* to other visual processes. Further we have shown that collated features are useful for the generation of shape descriptions and object segmentation.

For segmentation, one main advantage of our system, that we see over previous systems, is that it provides a shape description of the segmentations. This allows a much more systematic application of the segmented features.

RIGHT IMAGE LEFT IMAGE

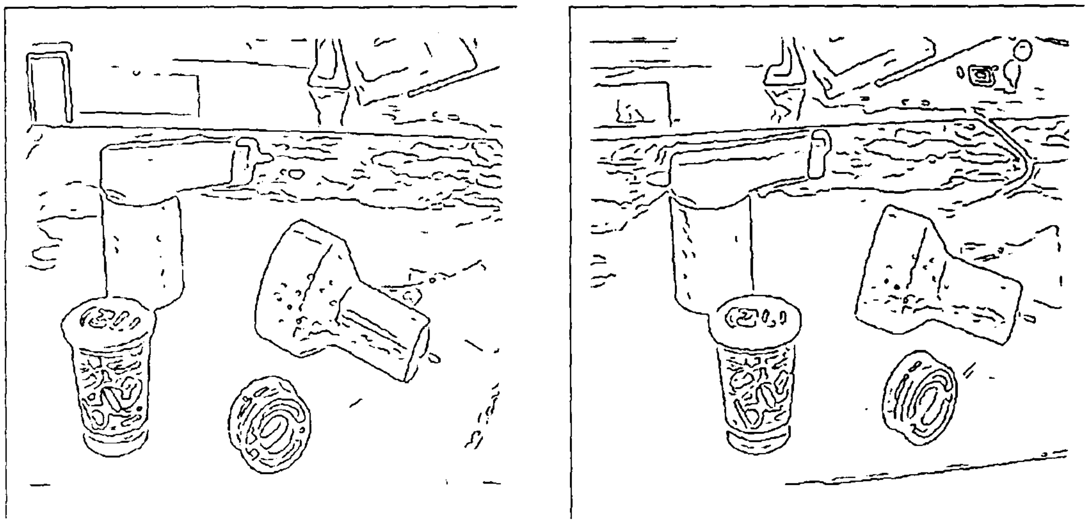


Figure 11: Edges detected in Stereo Pair

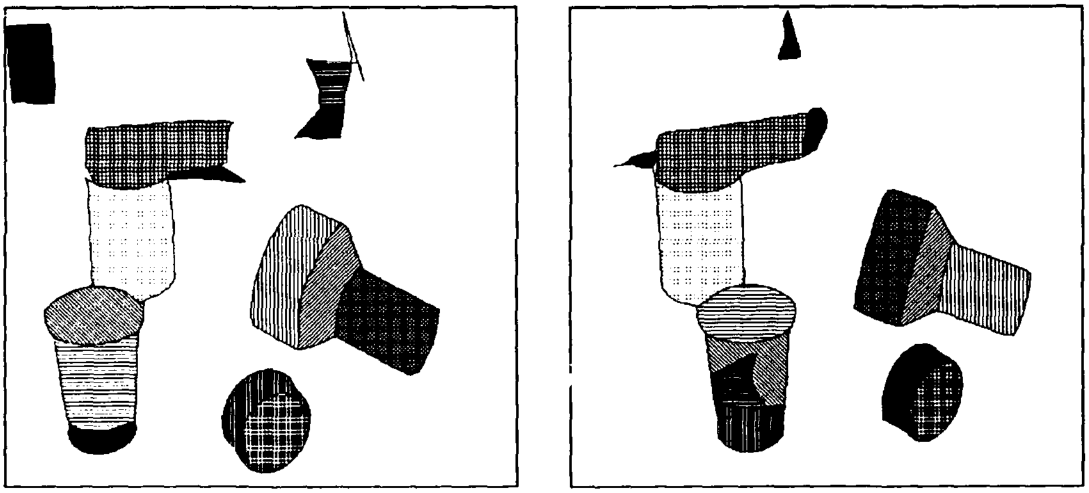


Figure 12: Ribbons detected in Stereo Pair

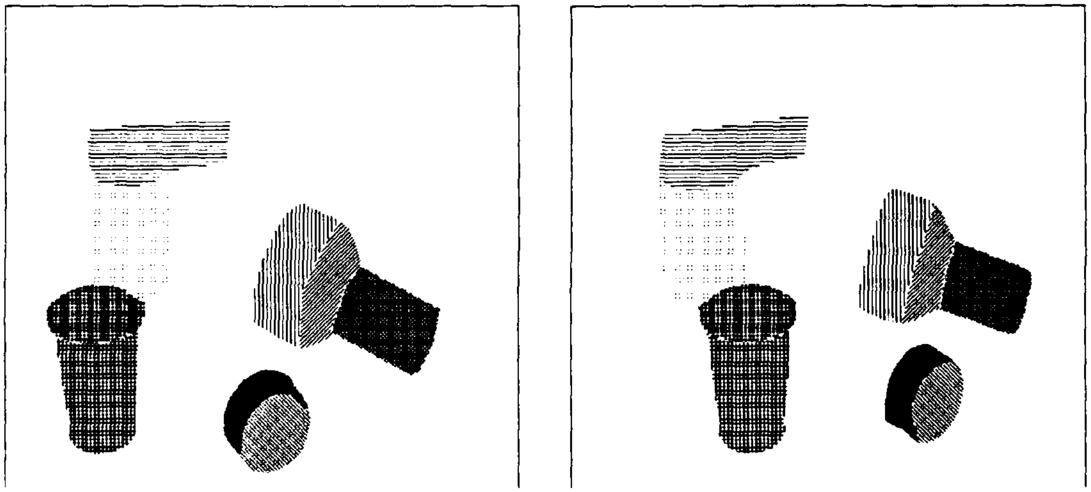


Figure 13: Ribbons matched in Stereo Pair

References

[1] D.G. Lowe. *Perceptual Organization and Visual Recognition.* Kulwer Academic Press, Hingham, MA, 1985.

[2] A.P. Witkin and J.M. Tenenbaum. On the Role of Structure in Vision, in *Human and Machine Vision*, Beck, Hope and Rosenfeld eds., pages 481–543. Academic Press, New York, NY, 1983.

[3] R. Mohan and R. Nevatia. Using perceptual organization to extract 3-D structures. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, to appear.

[4] D.J. Jacobs. GROPER: A grouping based recognition system for two dimensional objects. In *Proceedings of the IEEE Computer Society Workshop on Computer Vision*, Miami Beach, Florida, December 1987.

[5] T.O. Binford. Inferring surfaces from images. *Artificial Intelligence*, 17, 205–245, 1981.

[6] M.A. Fischler and R.C. Bolles. Perceptual organization and curve partioning. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, PAMI-8(1):100–105, January 1986.

[7] K.A. Stevens and A. Brookes. Detecting structures by symbolic constructions on tokens. *Computer Vision, Graphics and Image Processing*, 37:238–260, 1987.

[8] J.F. Canny. A computational approach to edge detection. *IEEE Transactions on Pattern Recognition and Machine Intelligence*, 8(6):679–698, November 1986.

[9] P. Saint-Marc and G. Medioni. Adaptive smoothing for feature extraction. In *Proceedings of the DARPA Image Understanding Workshop*, Boston, Massachusetts, April 1988.

[10] A. Rosenfeld. Axial representation of shape. *Computer Vision, Graphics, and Image Processing*, (33):156–173, 1986.

[11] T.O. Binford. Visual perception by computer. *IEEE Conference on Systems and Controls*, December 1971. Miami.

[12] G.J. Agin and T.O. Binford. Computer description of curved objects. *IEEE Transactions on Computers*, 25, April 1976.

[13] R. Nevatia and T.O. Binford. Description and recognition of complex-curved objects. *Artificial Intelligence*, 8:77–98, 1977.

[14] R. Nevatia. *Computer Analysis of Scenes of 3-D Curved Objects.* Birkhauser-Verlag, Basel, Switzerland, 1976.

[15] Michael Brady. Criteria for Representation of Shapes, in *Human and Machine Vision*, Beck, Hope and Rosenfeld eds., pages 39–84. Academic Press, New York, NY, 1983.

[16] J.H. Connell and M. Brady. Generating and generalizing models of visual objects. Technical Report AIM-823, Massachusetts Institute of Technology, July 1985.

[17] J. Ponce, D. Chelberg, and W. Mann. Analytical properties of generalized cylinders and their projections. In *Proceedings of the DARPA Image Understanding Workshop*, Los Angeles, CA, Feburary 1987. Morgan Kaufmann Publishers.

[18] T. Kanade. Recovery of the three-dimensional shape of an object from a single view. Technical Report CMU-CS-79-153, Carnegie-Mellon University, October 1979.

[19] F. Ulupinar and R. Nevatia. Using symmetries for analysis of shape from contour. In *Proceedings of the Internation Conference on Computer Vision*, Tampa, Florida, December 1988.

[20] J.J. Hopfiled and D.W. Tank. Computing with neural circuits: A model. *Science*, 233:625–633, 1986.

CONSTRAINT-BASED MODELING

J.L. Mundy, P. Vrobel, and R. Joynson
GE Corporate R&D Center
Schenectady NY 12301

ABSTRACT

An approach which combines symbolic and numerical methods for the solution of systems of geometric constraints is described. Such constraints arise from the description of parameterized object models as well as the geometric relationships between objects, cameras and light sources. Typical applications are environmental modeling for photointerpretation and autonomous navigation. A series of intial experiments to investigate the effect of symbolic variable elimination on numerical convergence with a commercial nonlinear programming package available in the IMSL library are described. Experimental issues such as the limit on problem complexity and machine architecture are also presented. A symbolic method for determining an appropriate choice for the independent parameters of a geometric configuration is discussed. Some initial theoretical results on the detection of singularities in the constraint system are also presented.

INTRODUCTION

Geometric models provide an effective representation for the recognition of objects and for the description of their environment [Lowe], [Huttenlocher and Ullman], [Lamdan et al], [Grimson and Lozano-Perez], [Thompson and Mundy]. The basic approach is to form a correct set of assignments between image features and model features. The feasibility of such assignments is typically determined by *viewpoint consistency* which means that all features of an object must project into an image with the same coordinate transformation. Thus, feature assignments are inconsistent if they produce different model-to-image transformation parameters. The location and orientation of objects in the environment is specified by the transformations determined from this model-matching process.

Much of the work cited above has focussed on polyhedral object models. A different model is constructed for each object to be recognized. This approach is reasonable for a small number of models, but some form of generalization is necessary in order to effectively represent a large library of objects. One approach to generalization is the parameterized object model such as the generalized cylinder [Brooks]. Another example is the super-quadric [Pentland]. With both of these schemes, a large number of object shapes can be represented with relatively few parameters.

In this paper we focus on the representation of object classes in terms of parameterized polyhedra. In our opinion, there are a number of potential advantages to be realized from this choice:

- Many applications, such as recognition, graphics and simulation, utilize polyhedral models. Recognition methods are much more advanced for polyhedral structures. Most engineering applications rely on polyhedral (finite element) structural models.

- Operations on polyhedral models, such as intersection and attachment, produce polyhedra as the result. Thus, a uniform topological representation can be maintained.

- The same representation can be used to represent geometric relationships between objects in the scene. For example, the relative orientations between buildings and between buildings and the ground plane can be incorporated within the model description as geometric relations on polyhedral faces and edges.

[1] Work at GE was supported in part by the DARPA Strategic Computing Vision Program in conjunction with the Army Engineer Topographic Laboratories under Contract No. DACA76-86-C-0007 and the Airforce Office of Scientific Research under contract No. F49620-89-C-003.

A major motivation for this work is acquisition of object models from image data and in particular, to acquire models for use in computer-assisted photointerpretation [Corby et al]. Parameterized polyhedra have already been used in modeling the environment for cartographic and navigational purposes [Hanson and Quam]. In SRI's Cartographic Workstation, various parameterized building shapes are available. The model parameters are determined manually by interactive parameter adjustment in the context of various image views of the site. The model parameters are adjusted so that the model image projections are consistent with the actual image data as well as with ground elevation data.

From a slightly more general viewpoint, it is desirable to determine an appropriate set of model parameters which are consistent with available image data and also consistent with a priori information about the scene. Examples of other constraints are:

- The relationship between buildings defined by map information.
- The relationship between objects and their shadows.
- Functional relationships. For example, aircraft are usually parked in a standard location and orientation.

Thus, the goal of the work to be presented here is the automatic determination of model parameters, or more generally the parameters defining the entire scene environment, in the context of multiple image views as well as other environmental constraints. Initially, we focus on the problem of automatically determining model parameters, given an arbitrary set of empirical constraints on the model components. We are taking both a theoretical and experimental approach.

PARAMETER DETERMINATION

REPRESENTATION FOR LINES AND PLANES

The first step is to define a representation for the model entities that is suitable for symbolic manipulation and requires a minimum number of parameters. In addition, we want to introduce the parameters in such a manner that we avoid the possibility of degenerate geometric configurations. For example, if a line is represented in terms of two points, the line is only defined when the points are unequal. Similarly, a plane becomes degenerate when defined in terms of three collinear points. Such conditions introduce inconvenient inequalities that must be maintained during parameter determination. We have been making investigations with three different parameterization schemes each of which have their own advantages and disadvantages for representing a model.

Point-Based Scheme Most constraint-based modeling systems use characteristic points on the model to constrain the geometry [Lin Gossard Light]. For a polygonal model, these characteristic points are the vertices. A model with N number of vertices requires 3N independent variables- an X, Y, and Z parameter for each point in space. All higher dimensioned entities are represented in terms of point parameters. The representation of a line is a vector in terms of six parameters- three parameters for each endpoint. The representation of a plane requires three non-colinear points which results in nine parameters.

Plane-Based Scheme In contrast to a point-based scheme, a plane-based scheme represents all lower dimensioned geometric elements in terms of plane parameters. A plane is represented in this parameterization with four independent variables. The variables include three components of the plane's normal vector and a parameter specifying the distance along the normal from the origin to the plane. Thus, a model with N number of faces requires 4N independent parameters. With this representation, a line is represented as the intersection of two non-parallel planes, and a vertex is represented as the intersection of three planes. In order to experiment with this scheme, we have limited our model test cases to a class of 2-manifold models with trihedral vertices. This restriction allows for a clean and consistent intersection representation.

Hierarchical Plane-Based Scheme The hierarchical plane-based scheme represents entities of lower dimension in terms of the next higher dimensional entity on which it lies. We proceed with an ordered hierarchy of definition for geometric entities: $\pi \succ \lambda \succ \rho$; where π, λ and ρ denote the classes of planes, lines and points, respectively. The ordering indicates that at least one plane surface is defined. Any line or point definition is made with respect so an existing planar coordinate system. Were appropriate, any point is defined with reference to an existing plane and a line lying in that plane.

A total of six parameters are required when a point is defined by first defining a plane and then a line and then the point. When a point is defined directly, only three parameters are needed. However, in the first case, a plane and a line are defined in addition to the point of interest and these extra entities are frequently used in other entity definitions. In any case, any extra parameters can be eliminated by solving incidence equations. This representation scheme always minimizes the number of parameters needed overall.

EMPIRICAL CONSTRAINTS

The central problem is to determine a set of numerical parameter assignments which are consistent with the constraints defined for the object model and its environment model, as well as empirical data such as image features that correspond to projections of the model. We wish to maintain all of the model geometric relationships, while at the same time approximating as closely as possible the observed image features and other empirical constraints.

The standard approach to this problem is to define a cost function which measures the mean square error between the measured model projections and the predicted values. The problem then becomes one of minimizing this cost function by adjusting the model parameters, subject to the model constraint equations. For example, consider the projection and image of a cube where the image features are not perfectly segmented. The parameters of the cube are adjusted to give the minimum error between the projected cube image and the empirical image features. The problem of minimizing a cost function subject to a nonlinear system of constraints is known as nonlinear programming [Luenberger].

The observation that model parameter determination is equivalent to nonlinear programming is hardly a solution to the problem. It is well known, that this problem is ill-behaved in general [Gill 1985]. For example, the application of these methods to the determination of camera parameters is not successful unless a good initial guess for the unknown camera parameters is available [Tsai]. Our goal here is to develop methods which are well behaved in the context of specific model configurations.

NONLINEAR PROGRAMMING

We consider that the parameter set of the object model is partitioned into independent and dependent parameters. The formal problem of parameter determination can be described within a nonlinear programming framework. We assume that a cost function, $f(\mathbf{u}, \mathbf{x})$, is defined. The variable set $\mathbf{u}$ represents the independent or free parameters of the system of constraints, $\mathbf{h}(\mathbf{u}, \mathbf{x})$. These independent variables are determined by minimizing the cost function subject to the system of constraints. $\mathbf{x}$ can be determined by solving the system of constraints.

The constraints are established algebraically by, $\mathbf{h} = 0$. That is,

$$
\begin{aligned}
h_1(u_1, u_2, \cdots u_{n-m}, x_1, x_2, \cdots x_m) &= 0 \\
h_2(u_1, u_2, \cdots u_{n-m}, x_1, x_2, \cdots x_m) &= 0 \\
&\cdots \\
h_m(u_1, u_2, \cdots u_{n-m}, x_1, x_2, \cdots x_m) &= 0
\end{aligned}
$$

There are numerous algorithms for determining the solution to this constrained minimization problem [Luenberger] [Gill et al]. The general approach is to find the gradient, $\nabla f(\mathbf{u}, \mathbf{x})$, and then incrementally move in the direction of decreasing $f(\mathbf{u}, \mathbf{x})$. A solution is declared when the cost function is at a local minimum and the constraints are satisfied.

In addition to the set of equality constraints, $\mathbf{h} = 0$, there can be constraints in the form of inequalities, $\mathbf{b}(\mathbf{u}, \mathbf{x}) > 0$. These inequalities naturally arise as geometric constraints which express relations such as, *between* and *above*. For example if the position of points, P_i, on a line are parameterized by t_i, then the inequalities,

$$\begin{aligned} b_1 &: \; t_2 - t_1 > 0 \\ b_2 &: \; t_3 - t_2 > 0 \end{aligned}$$

express the relation, $Between(P_1, P_2, P_3)$.

The inequalities are treated as a filter on the possible solutions of the minimization problem. That is, suppose that there are a number of solutions $\mathbf{u}_i$ which minimize $f(\mathbf{u}, \mathbf{x})$. The feasible solutions are those which also satisfy $\mathbf{b}(\mathbf{u}_i, \mathbf{x}) > 0$ An alternative approach to dealing with inequalities is described elsewhere in these proceedings [Cyrluk and Kapur].

Many nonlinear programming algorithms treat all of the parameters on an equal basis. For example, the method of Lagrange multipliers introduces the Lagrangian:

$$L = f(\mathbf{u}, \mathbf{x}) + \lambda^t \cdot \mathbf{h}(\mathbf{u}, \mathbf{x})$$

where λ is a set of m multipliers.

Then $(n + m)$ equations are generated for $(n + m)$ unknowns as follows:

$\nabla_{\mathbf{u}} L = 0$: (n - m) equations

$\nabla_{\mathbf{x}} L = 0$: m equations

$\nabla_{\lambda} L = 0$: m equations, the original constraints.

There is no distinction among the $(n + m)$ variables. Many different partitions of the variables may be produced by pivoting operations during the numerical solution process.

However, we choose to form a fixed partition of the variable set into a set of parameters attached to the constraint specification, the dependent parameters, and a set of degrees of freedom for the object. We conjecture that there is an advantage to be gained in convergence reliability by carefully selecting the degrees of freedom of the model configuration. By specifying the independent parameters, it becomes possible to analyze the convergence properties of the resulting constraint problem without reference to specific empirical data values.

The distinction between dependent and independent parameters is somewhat arbitrary, but the choice of $\mathbf{u}$ ideally satisfies the following requirements:

- $\mathbf{u}$ is consistent - the independent parameters are selected subject to the consistency of $\mathbf{h}$.
- $\mathbf{u}$ leads to convergence - the choice of independent parameters produces a reliable, convergent numerical minimization process.

The set of independent variables is considered to form a constraint surface, $\sigma(\mathbf{u})$ with dimension equal to the number of independent variables. The numerical minimization process is confined to this surface which is a subspace of the space of model parameters. The search for a minimum of $f(\mathbf{u}, \mathbf{x})$ on $\sigma(\mathbf{u})$ is known as the reduced gradient method [Luenberger]. One can view the constrained minimization process as equivalent to an unconstrained minimization in the subspace $\sigma(\mathbf{u})$. That is, the constraint equations can be used to determine $\mathbf{x}(\mathbf{u})$ and then the problem becomes one of unconstrained minimization with respect to $\mathbf{u}$.

In the case of nonlinear constraint equations, it is not easy to perform the elimination of the dependent variables. No closed form symbolic solution is possible for a single equation of greater than degree four. Equations of second degree and higher have multiple solutions, many of which may not correspond to feasible geometric configurations. Only real solutions to constraint equations can correspond to valid geometric configurations.

A method does exist to generate the real solutions for general symbolic polynomial systems, cylindrical algebraic decomposition [Arnon et al]. However, the algorithm involves computationally expensive polynomial resultant and root isolation procedures and so far has not solved any examples of practical interest.

However, it is straightforward to symbolically eliminate dependent parameters from linear constraint equations corresponding to the incidence between model elements. We have carried out experiments with this idea, as well as eliminating parameters from certain quadratic equations, and the results are discussed in a later section.

An alternative approach to variable elimination is to algebraically characterize the constraint surface in terms of singularities and specify well behaved paths for attaining the global minimum. The constraint surface can be complex, with folds and cusps. The goal is to reach the global minimum of $f(\mathbf{u},\mathbf{x})$ while maintaining the set of inequalities.

In the next section we consider the problem of determining a consistent set of independent parameters, followed by a section describing a symbolic method for analyzing the constraint surface.

DETERMINING THE INDEPENDENT PARAMETERS

The parameters associated with a particular geometric configuration are defined by the plane, line and point representations previously discussed. There is nothing in the representation that defines which parameters are to be considered dependent and which are independent. In fact, one must be careful in selecting independent parameters since it is easy to make a selection which is inconsistent with the model constraints. For example, if two lines are parallel, then the direction parameters of each line cannot be both independent.

The set of independent parameters should be such that the remaining dependent parameters can be consistently determined from the constraint equations. In most cases, m constraints on n unknown parameters leaves $n - m$ independent parameters. For nonlinear constraints, this relationship is not always valid, since some constraint equations may only serve to eliminate solutions of the remaining constraint set rather than reducing the number of independent parameters.

Fortunately, the general problem can be avoided since we are mainly interested in geometric constraints of incidence, orientation and distance. One can characterize these relationships and thus define a specific procedure for selecting a valid set of independent parameters. The reduction in the number of independent parameters for various relations on model entities is shown in Figure .

In this chart, the entities are assumed to be defined in a three dimensional coordinate frame. For example, the equality of two planes in space reduces the total number of parameters from 6 to 3. Likewise *the intersection of a point and a plane reduces the total number of parameters by one. A perpendicular* constraint between two lines reduces the number of independent parameters by one.

The effect of geometric relations on the model parameters can be represented as a network which we refer to as the *dependence network*. The elements of this network are defined in Figure . The nodes of the network represent parameters associated with the point, line and plane. These nodes are grouped about icons which represent each entity; a triangle for the plane, a square for the line and a circle for the point.

The directed edges of the network represent dependency between parameters. The edge orientation function, γ_{ij}, is defined by:

$$\gamma(e_{ij}) = +1 \text{ if parameter i depends on parameter j}$$
$$\gamma(e_{ij}) = -1 \text{ if parameter j depends on parameter i}$$

The edge direction is indicated by an arrow pointing away from the dependent parameter. We also define,

$$d_i = \sum_j \frac{1+\gamma_{ij}}{2}$$

There are three possible states for a node, i,

1) $d_i = 1$ - determined, the value of parameter i depends on other parameters.

2) $d_i = 0$ - the node is undetermined.

3) $d_i > 0$ - inconsistent, parameter i is defined in two or more ways.

Each entity and relation defines a disjoint subset, N_k, of the nodes. Associated with each N_k is an integer, Θ_k, which corresponds to the number of parameters needed to exactly determine the model entity or geometric relation. For the plane, line and point, Θ = 3,4 and 3, respectively. Each entity can also have a number of independent parameters, n_k. The sum of determined and independent parameters for each entity is always

	Point	Line	Plane
Point	3	-	-
Line	-	4	-
Plane	-	-	3

EQUAL

	Point	Line	Plane
Point	3	2	1
Line	2	1	2
Plane	1	2	0

INCIDENT

	Point	Line	Plane
Point	-	-	-
Line	-	2	1
Plane	-	1	2

PARALLEL

	Point	Line	Plane
Point	-	-	-
Line	-	1	2
Plane	-	2	1

PERPENDICULAR

Figure 1: The effect of various geometric relations on the degrees of freedom of model entities.

Θ_k. The examples in Figure illustrate several incidence constraints. The dependency network can be generated directly from the model specification.

The consistent assignment of edge directions can be achieved by the following linear program,

Minimize the number of independent parameters, N_{ind}, subject to:

$$1 \geq d_i \geq 0$$

$$n_k + \sum_{i \in N_k} d_i - \Theta_k = 0$$

This form is a special case of the network flow problem which can be efficiently solved [Sugihara][Luenberger]. There can be multiple solutions, all with the same number of independent parameters. In order to select among these solutions, the numerical convergence properties of a particular choice of independent parameters must be considered. This issue is discussed in the next section. More network examples are illustrated in Figure . A consistent assignment of edge directions is shown for the case of two perpendicular lines with equidistant constraints.

SYMBOLIC ANALYSIS OF CONVERGENCE

THE REDUCED GRADIENT METHOD

In the reduced gradient method of solving the nonlinear programming problem, the variable set is partitioned into independent and dependent variables, **u** and **x**, respectively [Luenberger]. One can view the constrained minimization process as equivalent to an unconstrained minimization in the subspace $\sigma(\mathbf{u})$. The reduced gradient in $\sigma(\mathbf{u})$ is given by,

$$\nabla^r_{\mathbf{u}} = \nabla_{\mathbf{u}} f(\mathbf{u}, \mathbf{x}) + \lambda^t \nabla_{\mathbf{u}} \mathbf{h}(\mathbf{u}, \mathbf{x})$$

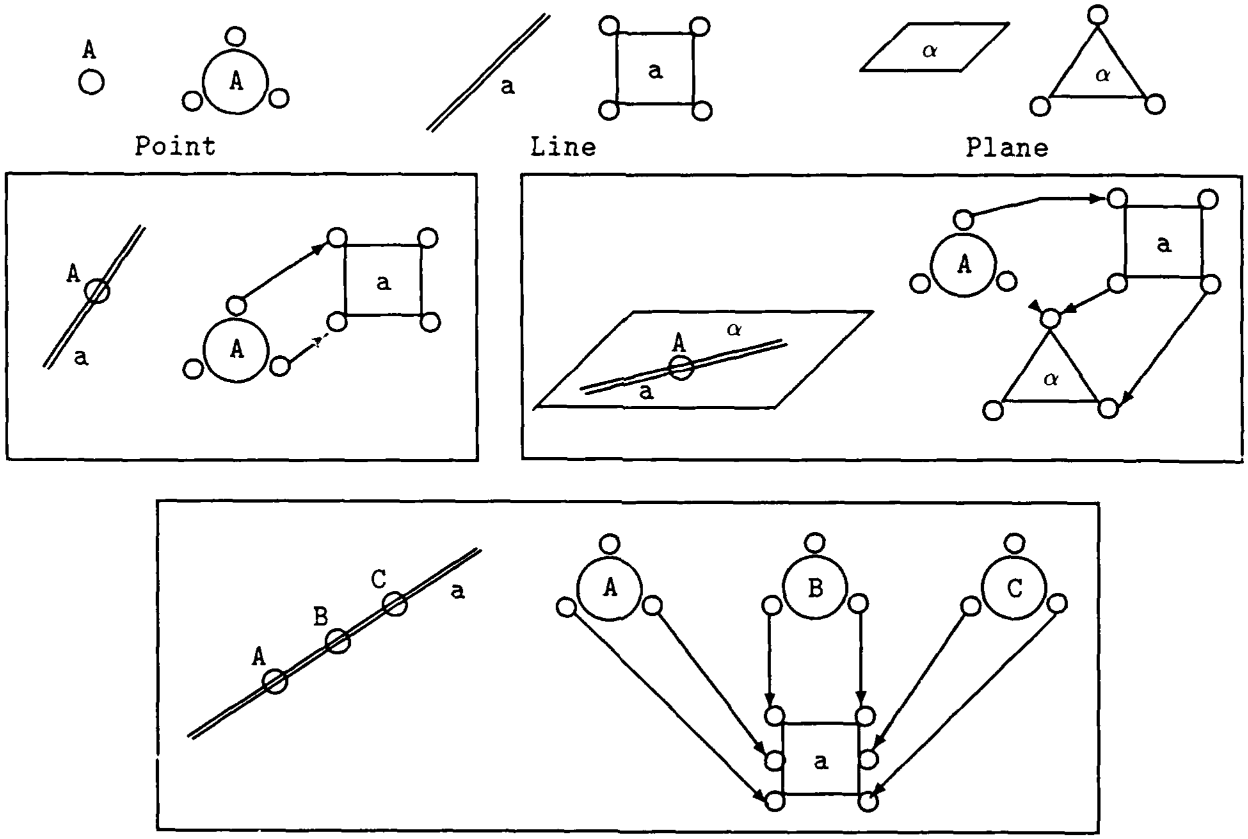


Figure 2: The elements of the parameter dependency network, along with several examples.

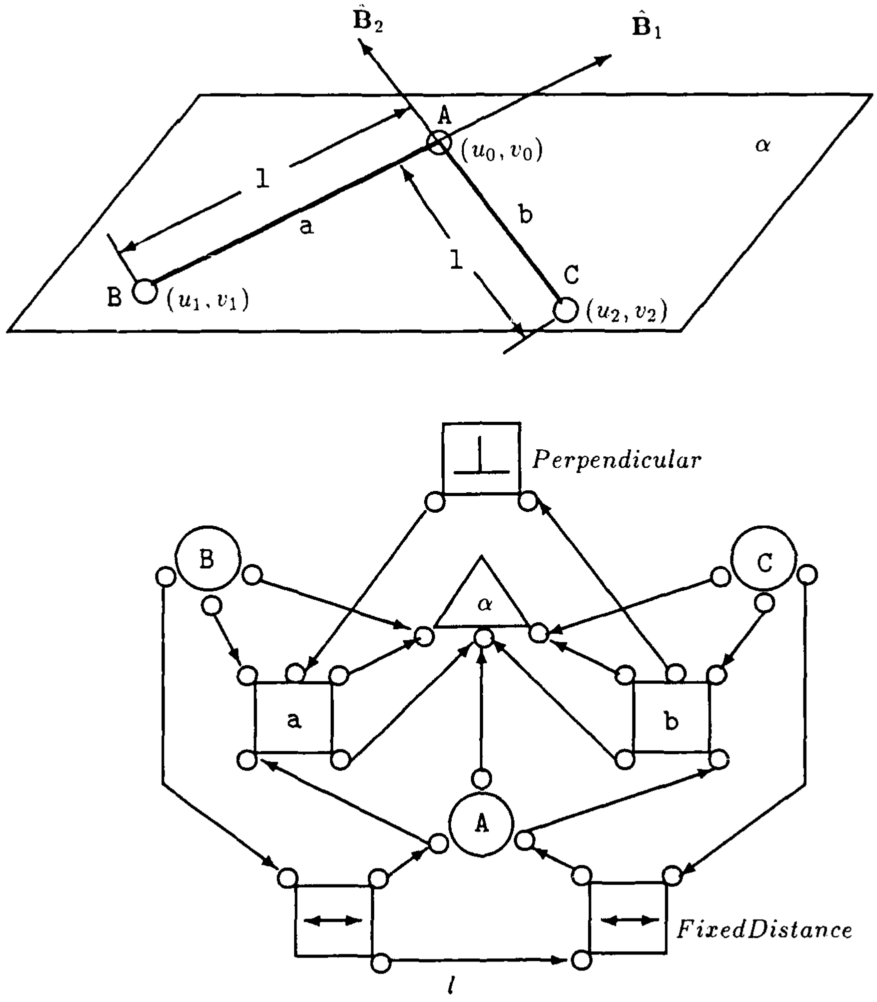


Figure 3: This example illustrates the use of perpendicular and distance constraints.

where $\nabla_{\mathbf{u}}$ indicates the gradient with respect to $\mathbf{u}$. A solution for the minimization of f is given by,

$$\nabla^{f}_{\mathbf{u}} = 0$$
$$\mathbf{h}(\mathbf{u}, \mathbf{x}) = 0$$

The Lagrange multipliers, λ satisfy the following equation,

$$\nabla_{\mathbf{x}} f(\mathbf{u}, \mathbf{x}) + \lambda^{t} \nabla_{\mathbf{x}} \mathbf{h}(\mathbf{u}, \mathbf{x}) = 0$$

So,

$$\lambda^{t} = -\nabla_{\mathbf{x}} f(\mathbf{u}, \mathbf{x}) \mathbf{J}_{h}^{-1}$$

where $\mathbf{J}_h$ is the Jacobian of the constraints with respect to the dependent variables. That is, $\mathbf{J}_h = \nabla_{\mathbf{x}} \mathbf{h}(\mathbf{u}, \mathbf{x})$. Thus, in order for a solution to exist at a point, $\sigma(\mathbf{u}^*)$, the Jacobian matrix must be nonsingular and $\mathbf{h}(\mathbf{u}^*, \mathbf{x}) = 0$.

The method for selecting the independent variables is as described in the previous section, so that the constraints are consistent. There is no guarantee, however, that the Jacobian matrix is everywhere nonsingular on $\sigma(\mathbf{u})$. Thus, we describe a method for determining the singularities of the constraint Jacobian, given a particular selection of independent variables.

THE TRIANGULATION METHOD

The singular points of $\mathbf{J}_h$ correspond to the vanishing of the determinant of $\mathbf{J}_h$, or $|\mathbf{J}_h| = 0$. The constraints, $\mathbf{h}(\mathbf{u}, \mathbf{x})$, can be put into triangular form by a symbolic procedure introduced by Ritt [Kapur and Mundy]. The triangulation process produces a new set of constraints, $\mathbf{g}(\mathbf{u}, \mathbf{x})$, of the form,

$$\begin{aligned}
\mathbf{g}_1(u_1, u_2, \cdots, u_{n-m}, x_1) &= 0 \\
\mathbf{g}_2(u_1, u_2, \cdots, u_{n-m}, x_1, x_2) &= 0 \\
\cdots \\
\mathbf{g}_m(u_1, u_2, \cdots, u_{n-m}, x_1, x_2, \cdots, x_m) &= 0
\end{aligned}$$

With this triangular form, $\mathbf{J}_g = \nabla_{\mathbf{x}} g(\mathbf{u}, \mathbf{x})$, is a lower trianglar matrix. The determinant of $\mathbf{J}_g$ is simply the product of the diagonal elements of $\mathbf{J}_g$,

$$|\mathbf{J}_h| = \prod_i \frac{\partial g_i}{\partial x_i}$$

Thus, the singularities of $\mathbf{J}_g$ on $\sigma(\mathbf{u})$ correspond to the zeros of the diagonal elements. This singularity condition is conservative, since the common zeroes of $\mathbf{h}$ are a subset of the common zeroes of $\mathbf{g}$ [Kapur and Mundy]. That is, $\sigma_h(\mathbf{u}) \subseteq \sigma_g(\mathbf{u})$. Thus, singularities found in the triangulated constraints may not be singularities of the actual constraints, but the converse will always hold. The advantage of the triangulated form is that the singularity conditions are directly represented as a product of factors.

To illustrate the process, consider two perpendicular line segments with equal length constraints. This configuration was shown earlier in Figure and the parameters are defined there. The line directions and point coordinates are represented in the plane coordinate space (U,V). The constraint equations are,

$$\begin{aligned}
h_1 &: \quad B_{u1}^2 + B_{v1}^2 - 1 \\
h_2 &: \quad B_{u2}^2 + B_{v2}^2 - 1 \\
h_3 &: \quad B_{u1} B_{u2} + B_{v1} B_{v2} \\
h_4 &: \quad (u_1 - u_0) B_{v1} + (v_1 - v_0) B_{u1} \\
h_5 &: \quad (u_1 - u_0)^2 + (v_1 - v_0)^2 - l^2 \\
h_6 &: \quad (u_2 - u_0) B_{v2} + (v_2 - v_0) B_{u} \\
h_7 &: \quad (u_2 - u_0)^2 + (v_2 - v_0)^2 - l^2
\end{aligned}$$

A set of independent parameters $[B_{u1}, l, u_0, v_0]$ is selected, as well as an ordering on the dependent parameters $[B_{v1} < B_{v2} < B_{u2} < v_1 < u_1 < v_2 < u_2]$. The equations are triangulated using an algorithm implemented within the GEOMETER[GEOMETER] system with the following result,

$$\begin{aligned}
g_1 &: \ B_{u1}^2 + B_{v1}^2 - 1 \\
g_2 &: \ B_{u1}^2 B_{v2}^2 + B_{v1}^2 B_{v2}^2 - B_{u1}^2 \\
g_3 &: \ B_{u1} B_{u2} + B_{v1} B_{v2} \\
g_4 &: \ B_{u1}^2 (v_1 - v_0)^2 + B_{v1}^2 (v_1 - v_0)^2 - B_{v1}^2 l^2 \\
g_5 &: \ (u_1 - u_0) B_{v1} + (v_1 - v_0) B_{u1} \\
g_6 &: \ B_{u2}^2 (v_2 - v_0)^2 + B_{v2}^2 (v_2 - v_0)^2 - B_{v2}^2 l^2 \\
g_7 &: \ (u_2 - u_0) B_{v2} + (v_2 - v_0) B_{u2}
\end{aligned}$$

The determinant of $\mathbf{J}_g$ is,

$$|\mathbf{J}_g| = 16 B_{u1}^5 B_{v1}^2 B_{v2}^4 (B_{u1}^2 + B_{v1}^2)(B_{u2}^2 + B_{v2}^2)(v_1 - v_0)(v_2 - v_0)$$

The critical singularities, $[(u_1 = u_0), (v_1 = v_0)]$ and $[(u_1 = u_0), (v_1 = v_0)]$, are contained in the zeroes of $|\mathbf{J}_g|$.

As another example, consider two planes at right angles and with the surface normal of one plane constrained to be at an angle, θ, with the positive z axis. The constraint equations are,

$$\begin{aligned}
h_1 &: \ n_{x1}^2 + n_{x2}^2 + n_{z1}^2 - 1 \\
h_2 &: \ n_{x2}^2 + n_{y2}^2 + (Cos^2(\theta) - 1) \\
h_3 &: \ n_{x1} n_{x2} + n_{y1} n_{y2} + Cos(\theta) n_{z1}
\end{aligned}$$

Selecting independent variables, $[n_{x1}, n_{y1}]$, and with ordering on dependent variables, $[n_{z1} < n_{x2} < n_{y2}]$, the triangulated form is,

$$\begin{aligned}
g_1 &: \ n_{x1}^2 + n_{x2}^2 + n_{z1}^2 - 1 \\
g_2 &: \ n_{y1}^2 n_{x2}^2 + (n_{x1} n_{x2} + Cos^2(\theta) n_{z1}) + (Cos(\theta)^2 - 1) n_{y1}^2 \\
g_3 &: \ n_{x1} n_{x2} + n_{y1} n_{y2} + Cos(\theta) n_{z1}
\end{aligned}$$

The determinant of $\mathbf{J}_g$ is,

$$4 n_{y1} n_{z1} (n_{x2} n_{y1}^2 + n_{x1} (nx1nx2 + Cos(\theta) n_{z1}))$$

The geometric significance of the last factor may be seen by
substituting for $Cos(\theta) n_{z1}$ from g_3. The factor becomes,

$$n_{y1} (n_{x2} n_{y1} - n_{x1} n_{y2})$$

Thus, the jacobian becomes singular when the projections of the surface normal of each plane onto the x-y plane are collinear.

We are currently investigating similar symbolic methods for determining the eigenvalues of the hessian matrices of the cost function and constraint equations. This second order information is needed to characterize the convergence rate. The singularities of the jacobian and the loss of positive definiteness of the hessian must be avoided in order to guarantee rapid and stable convergence.

EXPERIMENTS

Because the behavior of sophisticated numerical methods for nonlinear programming cannot be analyzed theoretically to a degree that convergence may be predicted [Ecker and Kupferschmid], we have been investigating the convergence of typical 3D modeling problems by taking an experimental approach. The following sections describe the approach and results of testing.

TEST METHOD

The general approach to experimentation is the production of a number of test cases from which one may make comparisons and speculations. For our work, the process of producing a test case consists of building an arbitrary model topology and specifying 3D geometrical constraints on its structure. For the most part, these constraints are dimensional constraints between faces, edges, and vertices. Equations representing the constraining relationships are produced from the symbolic representations of the entities. The constraints which define the model are compiled into a system of equations and the unknown parameters to the equations are solved for numerically. The solution is then propagated back through the equations to create a specific geometry for the model. The basic steps of the testing procedure follow in detail.

Parameterization and Creation of Model Topology

For each test, a parameterization scheme is selected and a model of the structure being constrained is created. There are many different schemes for parameterizing a polygonal model each with its own advantages and disadvantages. The difference between the schemes is in the selection of parameters that represent the model. We have concentrated our efforts on comparing the three schemes discussed earlier: point-based, plane-based, and hierarchical plane-based

Regardless of the parameterization scheme used, an actual 3D model topology is created for the test structure. The modeled structure provides initial value data for the independent parameters. It also serves as a vehicle for validating the resulting numerical solution of the nonlinear program constraining the geometry of the model. Test structures are created by specifying a sequence of incremental Euler operations [Mantyla] that construct the model. A specification file in the form of Lisp functions is then compiled to create an object instance.

Constraint Specification

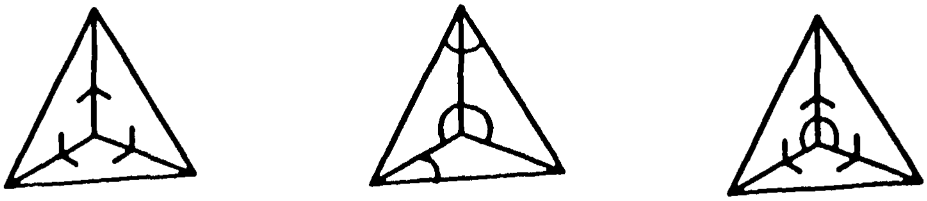
Once a test structure is created, the geometry of the structure is constrained. For our experiments thus far, we have been working with three different types of constraints: orientation, distance, and incident. At the present time, the ability to assign constraints to the geometry of a test structure is accomplished by a language specification in the same manner that an object topology is created. In addition, the language provides the mechanisms for construction of the symbolic representations for face, edge and vertex entities. Compilation of the constraints produces a file of equations that are in the form needed for input to the constraint solver.

Numerical Constraint Solving

Each test case is in the form of a nonlinear programming problem. For each constrained model problem, we would like a solution to be found that is close as possible to the initial unconstrained model geometry. To accomplish this, we use a cost function that is the sum of squared error between the independent parameters and their initial values computed from the unconstrained model. This objective function is then minimized subject to satisfying the geometrical constraints imposed on the structure.

Many commercial mathematical libraries exist today which contain functions for solving non-linear optimization problems. The algorithms represent years of research in numerical analysis and development experience in the behavior of ill-conditioning, degeneracy, and inconsistent constraints. Well constructed mathematical software is designed to deal with these types of problems [Gill 1985]. For the experiments discussed in this paper, the DNCONF function of the IMSL Math Library was utilized. This function uses a quadratic programming method to solve a general nonlinear programming problem posed in standard form. The algorithm is based on an iterative formulation and solution of quadratic programming subproblems. The subproblems are obtained by using a quadratic approximation of the Lagrangian and by linearizing the constraints [IMSL]. This algorithm uses a double precision finite difference method to compute gradients.

For each experiment, a constrained model is specified in mathematical format in a Fortran file and processed through the constraint solver of the IMSL library. The robustness of the constraint solving process has been explored using the IMSL routines on a CONVEX vector processor and a VAX 11/785. A speed



Model A: Corner Model B: Tetrahedron Model C: Corner

Figure 4 Constraint configuration on corner and tetrahedron test structures

comparison showed that the CONVEX ran a test constraint solving program in about 1/10 the time of the VAX As a result, all experiments have been conducted using the CONVEX to obtain this speed advantage

Verification

In order to compare results of convergence, three criteria are considered number of search iterations (both line search and overall gradient search), order of accuracy of the solution (how closely the constraint equations evaluate to zero), and value of the objective function (how closely the solution is to initial values) A practical verification is also performed by propagating the solution results back to the 3D model and computing a new geometry for the model This verifies that a solution is not a degenerate case.

TEST RESULTS

We have been conducting experiments to observe two characteristics of nonlinear numerical methods convergence and performance Convergence experiments involve changing the formulation of the nonlinear program and noting the effects on the numerical convergence Performance experiments exercise the numerical routines in such a way as to make measures on robustness In general, the test results described in this section are based on simple model structures including corners, tetrahedrons, triangular and rectangular prisms, and square pyramids The small problems that we have been dealing with have allowed us to observe the effect of incremental changes in the model structure more easily than more complex systems We will progress towards more complex models as we learn more about optimal problem formulation

Convergence Experiments

Most of the testing conducted thus far has been aimed towards determining the advantages of minimizing the number of parameters of a system representing a model. Choosing what parameters will be independent in a problem encompasses issues in selection of a parameterization for the model and elimination of dependent variables

Selection of Parameterization Scheme One method of minimizing the number of parameters in a problem is simply selecting a particular parameterization scheme which happens to formulate the system with the least number of independent variables. We have conducted simple comparison testing with a corner model and a tetrahedron using the three parameterization schemes discussed previously Figure 4 illustrates the three different perturbations of a simple problem set. The arrow notches in the figure represent perpendicularity constraints between the spanned faces The arcs represent equi length constraints between two edges of the structure

The Table 1, Table 2, and Table 3 show information about the numerical program and the results of the final convergence analysis The *Parameters* and the *Equations* columns represent the number of independent variables and the number of constraint equations respectively *Iterations* represent the number of times the

Scheme	*Parameters*	*Equations*	*Iterations*	*F(x)*	*Acc*
Point-Level	12	3	8	0.409436	10^{-9}
Plane-Level	9	3	9	0.212425	10^{-10}
Hier-Plane	6	3	5	0.022291	10^{-13}

Table 1: Convergence Results of Corner: Model A

Scheme	*Parameters*	*Equations*	*Iterations*	*F(x)*	*Acc*
Point-Level	12	6	6	36.9956	10^{-8}
Plane-Level	16	6	10	0.26570	10^{-11}
Hier-Plane	26	13	NC	-	-

Table 2: Convergence Results of Regular Tetrahedron: Model B

algorithm performed a gradient search before converging to a solution. The last two columns in the tables represent the value of the objective function, F(x), and the order of the least accurate constraint evaluation. NC in the *Iterations* column corresponds to No Convergence of the problem was obtained.

In each case, the parameter. ·ation with the least number of parameters resulted in the best convergence in terms of least number of gradient search iterations to the final solution. However, by comparing the values of the objective function and the order of accuracy, the fastest convergence does not necessarily guarantee the best quality solution. One might conclude that the reduction in the number of parameters helps to reduce the complexity of the nonlinear programming problem and it is a consideration in formulating the problem, but minimization of parameters is not a critical issue for insuring convergence. The non-convergence of the hierarchical plane-based testing in this experiment is believed to be related to issues involved with elimination of variables discussed in the next section.

Elimination of Variables The hierarchical plane-based parameterization scheme was developed as a minimal representation for a model. The representation makes wide use of eliminating parameters which would otherwise be independent variables in a problem set. This is accomplished by substituting into the constraint equations an equivalent form for the dependent parameters in terms of independent parameters. For example, the representation of the n_z component of a plane normal is substituted as $\sqrt{1 - n_x^2 - n_y^2}$. This form eliminates the use of n_z as an independent parameter. The B_v line direction parameter in plane coordinate space can be represented in the same manner as $\sqrt{1 - B_u^2}$ thus eliminating it from the free parameter set.

Consider the following model structure illustrated in Figure 5. This structure is constrained with three plane perpendicular constraints and one edge perpendicular constraint. The independent parameters associated with each constrained entity are shown. A sequence of experiments have been conducted with this model parameterized in the hierarchical plane-based scheme to determine whether elimination of variables using the assumptions of the representation is advantageous. Table 4 shows the results of some of the tests performed. Test 1 was conducted using all the dependent variable substitutions possible to eliminate parameters. In this case, $n0_z, n1_z, n2_z, n4_z, B00_v$ and $B01_v$ are made to be dependent variables. With this formulation, the numerical algorithm failed after 4 iterations claiming the gradient search was going uphill. A number of errors from taking the square root of a negative descriminent were also propagated during the process. The convergence problems encountered in Test 1 are conjectured to be a result of the square root form representing n_z. The n_x and n_y parameters iterate during the numerical process to values which result

Scheme	*Parameters*	*Equations*	*Iterations*	*F(x)*	*Acc*
Point-Level	12	5	8	0.419828	10^{-8}
Plane-Level	16	5	11	0.318751	10^{-7}
Hier-Plane	18	10	NC	-	-

Table 3: Convergence Results of Corner: Model C

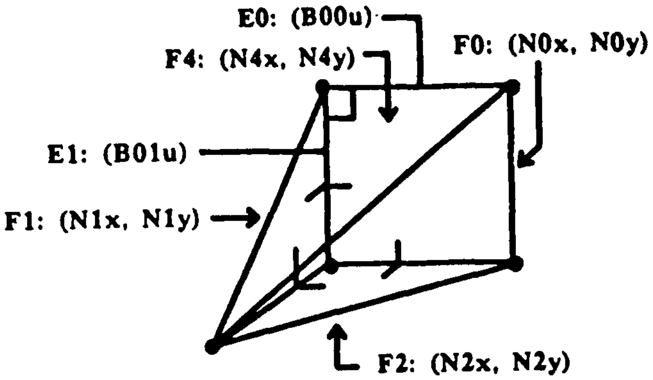


Figure 5: Constrained model structure of square pyramid.

Test	*Parameters*	*Equations*	*Iterations*	*F(x)*	*Acc*
1	10	10	NC	-	-
2	14	10	11	0.0263397	10^{-6}
3	16	10	31	0.017330059	10^{-9}
4	16	14	7	0.015637081	10^{-9}

Table 4: Convergence Results of Square Pyramid Tests.

in a negative descriminent of the square root term. In other words, the gradient search is following a path of convergence which goes through a problem area.

In order to eliminate the square root errors, the problem setups for Test 2 and Test 3 progressively added the n_z plane components as free variables and the B_v line components as free variables respectively. In order to represent line parameters in plane coordinate space, it is imperative that the normals of the planes be unit vectors. This constraint was previously implicit in the representation of n_z. In Test 2 the normals are explicitly normalized before being used to transform a line. In Test 3, the components of the line direction are explicitly normalized. In each case, the nonlinear program successfully converges and convergence is more accurate as more of the variables become independent.

Test 4 was conducted to determine whether the normalization of vectors would best be accomplished by constraints. In this case, four additional constraint equations for normalizing the plane normals were added to the problem set. As seen in Table 4, the overall convergence was not only dramatically faster, but more accurate than those cases where the normal was explicitly normalized.

Placement of Constraints In any interactive modeling environment, placement of constraints on the geometric entities of a model is dependent on the user's discretion. A few experiments were conducted with the tetrahedron model to observe any effects of constraint placement on convergence characteristics. The tetrahedron is specified with 5 Equi-length constraints. Figure 6 illustrates the two different constraint configurations considered. Model A illustrates the constraints being evenly distributed over all the edges of the model. Model B corresponds to focusing on a single edge to be involved with all constraints. Point-based parameterization tests consistently required 6 iterations to converge to a solution regardless of variations in configuration A or configuration B. Likewise,plane-based parameterization tests also showed consistent convergence characteristics requiring 11 iterations regardless of the configuration. In addition, each perturbation of the constraint configuration converged to the same solution.

Scaling Scaling of parameters is one of the common problem areas in solving nonlinear programming problems [Rice]. Using the tetrahedron model again, we have conducted scaling experiments where this problem becomes apparent. This test consisted of incrementally increasing the length specification of an edge in the problem for point-based and plane-based parameterization schemes. Some of the results are shown in Table 5.

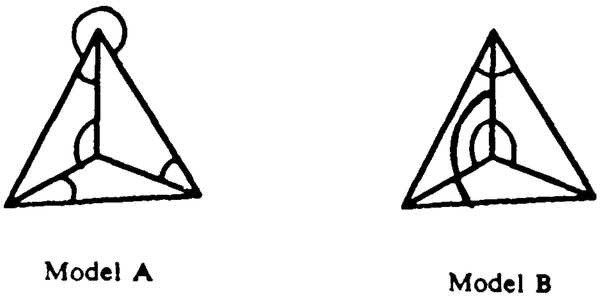


Figure 6: Two constraint distributions over a tetrahedron model.

Point-Based		*Plane-Based*	
Edge Length	*Iterations*	*Edge Length*	*Iterations*
0.01	12	0.01	15
1.414	6	1.414	11
-	-	5.477	21
-	-	10	28
-	-	11.18	467
22.36	6	22.36	NC
316.22	14	316.22	NC

Table 5: Results of iterative scaling examples.

The plane-based parameterization scheme includes plane normal components and a distance component in its independent parameter set. The normal component values are bounded within the range [-1,1] while the distance parameter can vary from $[-\infty,\infty]$. This difference in scale becomes a factor very quickly as can be seen with the sharp increasing trend in convergence of the plane-based tests. In contrast, the point-based scheme utilizes point parameters–all of which have identical value ranges. A point-based scheme would be the best representation for model structures which have dimension specifications that are of order 10 or greater.

Accuracy The output of an object recognition and positioning program sometimes generates inferred matches producing object orientations that are slightly askew. It was the purpose of this experiment to examine a means of "rectifying" the object by aligning it with other boundaries or surfaces whose positions and orientations are already known. A simple test involved orienting a misaligned cube into a particular position on a specified plane.

In order to construct a cube object, the vertices of the cube were constrained to fixed points by springs. The constraints, in this case, involved relations between the normals of the planes, i.e., the dot products of the normals of adjacent sides of the cube were zero. These, along with constraints setting the edge lengths, constituted the geometrical description of the object. Using just the geometrical constraints, the convergence was such that the resulting structure was a cube with vertices pointed toward their respective fixed points thus satisfying the minimizing function. Further testing with random perturbations of the fixed point locations demonstrated robust convergence behavior.

A positioning constraint is one which orients the cube with respect to its environment. This was applied by making the normals of two adjacent planes perpendicular to fixed directions. One may now rotate the cube by some rotation, say, tipping it. The initial alignment trials were often plagued by apparent convergence to a valid answer, by ending with either a "too many iterations in line search" error or an "attempting to climb uphill" error. A variant of the original IMSL routine allowed a convergence accuracy parameter to be set, rather than defaulted to an IMSL internal value. Pertubation of this parameter allowed the alignment

Rods	*Parameters*	*Single Precision*		*Double Precision*	
		VAX	*CONVEX*	*VAX*	*CONVEX*
1	6	0.13	0.02	0.20	0.03
10	60	10.3	1.3	17.7	1.5
20	120	64.7	7.2	112.8	8.5
30	180	184.2	19.3	335.2	22.5
36	216	338.8	34.8	657.9	40.8

Table 6: Results of timing tests in cpu seconds for rod experiments.

test to converge.

In general, most tests have converged when the constraint equations are solved to the order of 10_{-6} or better. We have had many experiences where the order of accuracy is better than 10_{-6}, but the convergence test within the numerical process does not stop the iteration. This experiment showed that by adjusting the accuracy threshold, a higher success rate in convergence is possible.

Performance Experiments

A few experiments have been conducted to explore the practical aspects of solving a system of constraints. When the decision was made to explore the use of numerical methods to solve the nonlinear optimization problems of 3D modeling, a number of questions arose:

1. What is the mechanism of entering a problem, i.e., the formulation of the problem?
2. Is there a limit to the number of variables that can be handled by the available software?
3. What times are involved in the solution of a problem of some complexity?
4. What strategies, i.e., use of double rather than single precision, might have to be employed to ensure convergence?

The Fortran routines of IMSL were available on two machines as explained above. In doing comparison testing, a simple problem that could easily be expanded to cover a wide range of variables was selected to help answer these practical concerns. The problem chosen was to simulate the suspension of a rod in 3-space with springs attached from the ends of the rod to fixed points in space. The cost function, or the function to be minimized, becomes:

$$f(x) = \sum_{i=1}^{6}(x_i - xg_i)^2$$

where x_i, i = 1,2,3 are the coordinates of a rod end, and x_i, i = 4,5,6 are the coordinates of the other end. Xg_i are the coordinates of the corresponding fixed points, or 'guesses at a solution'.

The constraint consists of specifying the length (= len) of the rod, and takes the form:

$$g(x) = (x_1 - x_4)^2 + (x_2 - x_5)^2 + (x_3 - x_6)^2 - len$$

With a single rod, the result is very simple. The rod extends to be of length len, and it is aligned along a line joining the fixed points with its mid-point coincident with the mid-point of the line. Convergence was obtained for all fixed point locations tried.

This model is very easily extended to a multiplicity of rods by respecifying the indices of the x's and xg's so that the evaluation of the constraints can be done in a loop, and supplying the appropriate limits. It does not matter that the rods are all in the same place. They are independent of one another in the formulation.

Using timing functions available in IMSL, the times of execution for the number of rods varying from 1 to 36, or 6 to 216 variables, were determined. Exerpts from the data are given in Table .

For a given machine and precision, the time varies as the cube of the number of variables. Running in single precision the times for the VAX are about 10 times that of the CONVEX, and in double precision, the factor is more like 16. A further observation is that the penalty for using double precision on the VAX doubles the time, whereas on the CONVEX a small increase of time is experienced. All subsequent use of IMSL was done on the CONVEX using double precision.

References

[Arnon et al] D.S. Arnon, G.E. Collins and S. McCallum, Cylindrical Algebraic Decomposition I, II, *SIAM J. of Computing*, Vol. 13, p.865 (1984)

[Brooks] R. Brooks, Symbolic Reasoning Among 3-D Models and 2-D Images, in *Computer Vision*, J.M. Brady Ed. North Holland (1981)

[GEOMETER] Connolly, C., D. Kapur, J.L. Mundy, R. Weiss, GEOMETER: A System for Modeling and Algebraic Manipulation, DARPA Image Understanding Workshop Proceedings, 1989.

[Corby et al] N.R. Corby, J.L. Mundy, P.A. Vrobel, A.J. Hanson, L.H. Quam, G.B. Smith and T.M. Strat, PACE an Environment for Intelligence Analysis, *Proc. DARPA IU Workshop*, p. 342, (1988)

[Cyrluk and Kapur] D. Cyrluk and D. Kapur, Solving Nonlinear Inequality Constraints: A Mulilevel Approach, DARPA Image Understanding Workshop Proceedings, 1989.

[Ecker and Kupferschmid] Ecker, J.G., M., Kupferschmid,*Introduction to Operations Research*, John Wiley and Sons, NY, 1988.

[Gill et al] P.E. Gill, W. Murray and M.H. Wright, *Practical Optimization* (1981)

[Gill 1985] Gill, P.E., W. Murray, M.A. Saunders, M.H., Wright, Model Building and Practical Aspects of Nonlinear Programming, *Computational Mathematical Programming*, K., Schittkowski (Ed.), Computer and Systems Sciences Vol. 15, Springer-Verlag, NY, 1985.

[Grimson and Lozano-Perez] W.E.L. Grimson and T. Lozano-Perez, Search and Sensing Strategies for Recognition and Localization of Two and Three Dimensional Objects, *Proc 3rd International Symposium on Robotics Research*, (1985)

[Hanson and Quam] A.J. Hanson and L.H. Quam, Overview of the SRI Cartographic Modeling Environment,*Proc. DARPA IU Workshop*, p. 576 (1988)

[Huttenlocher and Ullman] D.P. Huttenlocher and S. Ullman, Object Recognition Using Alignment, *Proc. 1st Int. Conf. on Computer Vision*, p. 102 (1987).

[IMSL] IMSL Math/Library: Fortran Subroutines for Mathematical Applications, Version 1.0, 1983.

[Kapur and Mundy] D. Kapur and J.L. Mundy *Special Issue on Geometric Reasoning of Artificial Intelligence*, Vol. 37, p. 21 (1988)

[Klaus] K. Schittkowski (Ed.), *Computational Mathematical Programming*, K., Schittkowski (Ed.), Computer and Systems Sciences Vol. 15, Springer-Verlag, NY, 1985.

[Lamdan et al] Y. Lamdan, J.T. Schwartz, H.J. Wolfson, Object Recognition by Affine Invariant Matching, *Proc. CVPR*, p. 335 (1988)

[Lin Gossard Light] Variational Geometry in Computer-Aided Design, Lin, V. C. , D. C. Gossard, R. A. Light, ACM Computer Graphics, Volume 15, Number 3, 1981, pp 171-179.

[Lowe] D.Lowe, *Perceptual Organization and Visual Recognition*, Kluwer Academic Publishers, Boston, MA, (1985).

[Luenberger] D.G. Luenberger, *Linear and Non-Linear Programming, 2nd Edition*, Addison-Wesley (1984)

[Mantyla] Mantyla, M., *An Introduction to Solid Modeling*, Computer Science Press, 1988.

[Pentland] A. Pentland, Perceptual Oganization and the Representation of Natural Form, *Artificial Intelligence Journal*, Vol. 28, No. 2, p.1 (1986)

[Rice] Rice, J. R., *Numerical Methods, Software, and Analysis: IMSL Reference Edition*, McGraw-Hill Book Company, NY, 1983.

[Sugihara] K. Sugihara, *Machine Perception of Line Drawings*, MIT Press (1986)

[Thompson and Mundy] D.W. Thompson and J.L. Mundy, Three Dimensional Model Matching From an Unconstrained Viewpoint, *Proc. IEEE Conference on Robotics and Automation*, p. 280 (1987)

[Tsai] R.Y. Tsai, An Efficient and Accurate Camera Calibration Technique for 3D Machine Vision, *Proc. CVPR*, p.364 (1986)

Acknowledgements The assistance of Prof. J. Michael Brady and the RRG research and administrative staff at Oxford is gratefully acknowledged.

Objective Functions for Feature Discrimination: Theory *

Pascal Fua Andrew J. Hanson

Artificial Intelligence Center
SRI International
333 Ravenswood Ave.
Menlo Park, CA 94025

Abstract

We propose and evaluate a class of objective functions that rank hypotheses for feature labels. Our approach takes into account the representation cost and quality of the shapes themselves, and balances the geometric requirements against the photometric evidence. This balance is essential for any system using underconstrained or generic feature models. We introduce examples of specific models allowing the actual computation of the terms in the objective function, and show how this framework leads naturally to control parameters that have a clear semantic meaning. We illustrate the properties of our objective functions on synthetic and real images. More details of the applications of the method are given in a companion paper.

Introduction

All approaches to the problem of extracting features from images can in principle be phrased in terms of decision theory; however, the concepts of decision theory are very hard to put into practice because of the difficulty of evaluating the required probability measures. Therefore, most practical approaches to model-based vision for both specific models [2,3,4,27] and generic models [8,21,20,18,19] rely on heuristic measures to select among competing scene parses. These methods, although they may be effective in the context for which they were designed, are extremely hard to extend and require the use of many parameters whose significance is not clearly understood.

On the other hand, approaches such as those of Feldman and Yakimovsky [7], Georgeff and Wallace [11], and Rissanen [22,23] provide a sound theoretical basis for the decision problem but offer few practical computational methods for dealing with complex scenes in real images.

In this paper, we draw a clear distinction between the generation of scene-labeling hypotheses and the mechanism by which the hypotheses are ranked; we focus on an objective function approach for the ranking task alone. We define a class of objective functions based upon theoretical arguments similar to those of Georgeff, Wallace and Rissanen [11,22,23], and show that the required probability estimates can actually be computed in the context of a few natural assumptions.

The approach suggests the definition of a minimal set of two parameters balancing the contributions of area photometry, edge photometry, and geometry. This balance allows us to establish a general context in which to understand generic edge-based methods such as those of Huertas and Nevatia [19], the region-based work of investigators such as Ohta, et al. [20], and hybrid approaches like that of McKeown and Denlinger [18].

A companion paper in these proceedings [10] illustrates the application of the objective function approach to both an operator-guided shape-refinement problem and a fully automated system for the extraction of buildings from aerial imagery. The interactive system deforms the contour of a user-supplied rough sketch to maximize the objective function of a specific model following the general paradigm proposed by Terzopoulos, Kass, and Witkin [28]. The automated system uses the basic tools of the interactive system, combined with heuristic rules that exploit appropriate components of the objective functions to generate a selected set of model hypotheses; the total objective function is applied to obtain a ranking of the resultant hypotheses.

*This research was supported in part by the Defense Advanced Research Projects Agency under Contract Nos. MDA903-86-C-0084 and DACA76-85-C-0004.

Our formulation has many desirable features, but is not by itself a complete solution to the feature extraction problem. To be effective it must be coupled with a robust hypothesis generation mechanism and an efficient optimization procedure. Furthermore, the simple examples of geometric quality analysis given below would benefit greatly from a comprehensive cognitive theory of shape perception. It should come as no surprise that *modeling* is the most difficult aspect of a system attempting to perform shape perception. Nevertheless, our approach provides a unified framework that clearly exposes the critical components and characteristics of model-based vision systems.

Derivation of the Objective Function

The goal of feature extraction is to parse a scene in terms of objects conforming to particular models. To discriminate among competing parses, an objective function must be able to measure the goodness of fit to feature models that include such characteristics as area photometry, edge photometry, shape, and semantic relationships. In this section, we define a basic class of models, discuss the parameters we expect to control our objective functions, derive the theoretical forms of the objective functions themselves, and provide an interpretation of the resulting functions in terms of information encoding theory.

Object Modeling

For the purposes of this work, we define a *model* to be a geometric description of an object in the world characterized by its *geometric constraints* and its *photometric signature*; we define the *evidence* for such objects in digital images to be a collection of *delineatable areas* corresponding to major object parts, together with associated quantities directly derivable from the pixel values in such areas.

We interpret the photometric signature of any object model in terms of the expected signal from *an ideal object model* plus *a noise model* [22,23,15]. The object's evidence can then be encoded in terms of these models. We will use length of the shortest encoding to measure the quality of the fit between the data and the model.

This division of the model language into object model plus noise is potentially task-dependent and semantic in nature. For example, if we are interested in *roofs*, we may consider the precise distribution of shingles on the roof to be irrelevant statistical noise; if we are interested in *shingles*, the position of each shingle on the roof becomes critical information. Textured object surfaces may similarly be either important in every detail or irrelevant except for their statistical character.

Essential Parameters of the Objective Function

Our approach introduces two fundamental parameters, the *scale* and the *shape coefficient*:

Scale. The scale is interpretable as the unavoidable dimensional factor that converts dimensional quantities such as area or length into dimensionless probabilities. Area units are thus scaled down by two powers of the dimensional unit, while length terms such as edges are scaled down by a single power. The scale parameter thus controls whether the area signature dominates edge signature.

The scale parameter may also be understood by observing that when an image is resampled or zoomed, the area A of a patch will change, but the complexity of the patch, as reflected in its minimal encoding, should remain invariant. Thus there should be some intrinsic zoom factor s that relates the area A to the area $A_0 = A/s^2$ in the zoomed image that has exactly the resolution needed to encode the model complexity without oversampling. The formulas presented later in the paper may thus be alternatively interpreted as expressing the patch encoding cost in terms of the sampling-invariant quantity A_0 instead of A itself.

In Appendix C, we suggest yet another way of understanding the scale in terms ot the minimal sampling rate needed to describe the image and its relationship to the Nyquist frequency.

Shape Coefficient. An objective function with a shape quality term alone will simply hallucinate its best model wherever it looks. An objective function with only a photometric model is equivalent to a segmentation algorithm [15]. The shape coefficient balances the possibly conflicting requirements of the geometry and photometry; the point where this balance lies must be determined by the context of the application.

The scale and shape coefficients characterize the fundamental balance of influences that must be semantically specified for each application. Within a particular model domain, it seems possible in principle to estimate the scale by using measures of local complexity. Our approach to feature-hypothesis evaluation provides a clear way to justify and understand the essential role of these two parameters in feature extraction, regardless of the other details of a particular system.

The Probability of a Scene Parse

We choose to describe the problem of determining the best image interpretation as the need to maximize the probability $P = p(m_0, m_1, \ldots, m_n|e_1, \ldots, e_n)$ that, given the evidence $E = \{e_i;\ i = 1 \ldots n\}$, parsing the scene in terms of a particular set of model instances $M = \{m_i;\ i = 1 \ldots n\}$ and a backround m_0 is in fact correct.[1] Each m_i is taken to be a geometric object model, while e_i is the measurable evidence for the object, typically a collection of associated pixel intensities. Since we are interested in feature extraction, we do not explicitly represent the background and collect no evidence for it.

It is essentially impossible to evaluate the conditional probability P in its most general form, so we make a crucial independence assumption: the probability of a particular model hypothesis is influenced *only* by its corresponding body of evidence and the other model instances. For example, in an aerial image, whether or not a patch of pixels can be identified as a road may depend on its own photometry and on the presence or absence of neighboring houses, but not on the particular photometry of those houses.

Formally, this assumption can be written as follows: If I, J, K denote sets of indices referring to model instances and their corresponding bodies of evidence, we assume $\forall I, J, K$ such that $J \cap I = \emptyset$ and $J \cap K = \emptyset$, $P(m_J e_K|e_I) = P(m_J e_K)$, and $\forall I, J,\ \ P(m_J|m_I, e_I) = P(m_J|m_I)$.

The assumption may break down when one object's expected photometry is strongly modified by another object, as when a superstructure or a separate building occludes or casts a shadow on a roof. In practice, one can partially compensate for such phenomena by discounting small anomalies.

Combining our assumption with Bayes' rule, it is straightforward to express the probability of the parse as:

$$P = p(m_0, m_1, \ldots, m_n|e_1, \ldots, e_n) = p(m_0, m_1, \ldots, m_n) \prod_{i=1}^{n} \frac{p(e_i|m_i)}{p(e_i)}. \tag{1}$$

This expression clearly separates the contribution of the photometry, in the evidence-dependent terms, from the abstract contribution of the geometric and semantic component in $p(m_0, m_1, \ldots, m_n)$ under the stated assumption. We further expand this term as:

$$\begin{aligned} p(m_0, m_1, \ldots, m_n) &= p(m_0|m_1, \ldots, m_n)p(m_1, \ldots, m_n) & (2)\\ &= P_0 p(m_1, \ldots, m_n), & (3) \end{aligned}$$

where $p(m_1, \ldots, m_n)$ is the probability that these n models appear in the scene, and P_0 is the probability that no other models appear. Since we do not take the background explicitly into account in this work, we consider P_0 to be constant. The details of the derivation are given in Appendices A and B.

Minimal Encoding Length and Model Effectiveness

We choose to express the quality of a parse as the (base 2) logarithm [2] of Eq. (1). As discussed in Appendix C, classical information theory [26,12] leads us to interpret the resulting score S in terms of encoding length:

$$S = +\log \frac{P}{P_0} = F - G, \tag{4}$$

where we define

$$\begin{aligned} F &= \sum_{i=1} F_i = \sum_{i=1} \{-\log p(e_i) + \log p(e_i|m_i)\} & (5)\\ G &= -\log p(m_1, \ldots, m_n). & (6) \end{aligned}$$

[1] For example, in terms of a human analyst's perception, or in terms of ground truth.

[2] All logarithms in this paper are base 2 logarithms.

Here F is what we call the *encoding-effectiveness* of the set of models. The first term in F is the number of bits needed to describe the evidence in the *absence* of the model, while the second term gives the number of bits needed to describe the evidence *in terms of the model.* The term *effectiveness* is thus motivated by the fact that F represents the *number of bits saved* by representing the evidence using the model, and that F increases as the fit improves.

G is the number of bits needed to encode the evidence-free model representation information, and quantifies the elegance of the chosen set of model instances as well as their dependencies.

Remarks

Feature Extraction Viewed as an Optimization Problem. The problem of finding the best parse of a scene can now be rephrased as the problem of optimizing over sets of hypotheses evaluated by Eq. (4). Global optimization corresponds to a blind search procedure, which searches all possibilities without attempting to determine which candidates are more likely than others. In practice, the search space may be far too large for this type of search. Since intelligent heuristics can overcome this drawback, a natural way to design an application system is to incorporate hypothesis-generation algorithms that *project* from the space of all possible hypotheses onto a subspace of very likely hypotheses [10]. Such projections have the side effect of reducing the discriminatory burden placed upon the objective function.

Equation (4), in contrast to formulations that attempt an exhaustive explanation of each pixel in a scene in terms of a minimal encoding framework [15], includes only features in the image, and thus describes a class of optimization problem that is better adapted to the feature extraction domain. In the examples presented, we do not attempt to encode the background, but only the foreground; the edge signature, discussed below, substitutes for the backgruund by measuring local foreground-background contrast.

Balancing the Evidence. When we consider $\{e_i\}$ to consist only of photometric evidence, Eq. (4) expresses the sought-for balance between the *photometric* and the *geometric* evidence supporting a model hypothesis. When the geometric information reflected in G is irrelevant or absent, the objective function reduces to a computation of maximum likelihood in the presence of the image-based information. When the evidence reflected in F is absent, the objective function evaluates the abstract geometric elegance of a particular parse.

Generic Models Require Photometric/Geometric Balance. When a model's geometry is completely determined beforehand, as it is for template-matching approaches to automatic shape recognition, there is *no need* for the geometric information component of the objective function, since it is constant and maximum likelihood analysis alone will do. The geometric terms in the model evaluation function begin to play a critical role when we utilize models defined by a general set of geometric constraints in place of a specific shape template. Such *generic models*, with arbitrarily large numbers of parameters, require objective functions like ours that balance their geometric aspects against their photometry.

Photometric Measures: Computing F

Two of the main characteristics of an object in an image are its interior photometry and its contrast with the background, which produces edges. Here we explore simple models for the area and for the edges of an object that have proven useful in analyzing imagery. When working with stereo pairs of images, we also incorporate a stereoscopic model, and compute the depth parameters of an object in the scene by optimizing the corresponding stereo effectiveness.

We have seen that the effectiveness F is computed as $-\log p(e) + \log p(e|m)$ where e represents the grey level values of the pixels that are enclosed by the contour m. For the sake of exposition, let us distinguish the evidence e_A relative to the interior of the patch and the evidence e_E relative to the boundary. Formally, we can write:

$$\begin{aligned} p(e|m) &= p(e_A|m)p(e_E|m, e_A) \\ p(e) &= p(e_A)p(e_E|e_A) \quad . \end{aligned}$$

We assume that contrast with the background can be measured by using local image derivatives, while ignoring the grey levels of the boundary pixels. This contrast depends on the grey level of background pixels that do not appear in the object descriptions, and can therefore be considered as independent of the interior object photometry. Thus we write F_i in Eq. (5) as the sum of area and edge components:

$$\begin{aligned} F_i &= F_{i,A} + F_{i,E} \\ F_{i,A} &= -\log p(e_A) + \log p(e_A|m) \\ F_{i,E} &= -\log p(e_E) + \log p(e_E|m) \quad . \end{aligned}$$

This prescription must be modified when dealing with objects that share edges, since the contrast of the shared edges is completely determined by the photometry of the regions on both sides of the edge. In this case, the shared boundaries do not contribute to the edge effectiveness term.

When additional images are available and m is a three-dimensional model, additional evidence e_S can be gathered using the projection of m onto each image. We write:

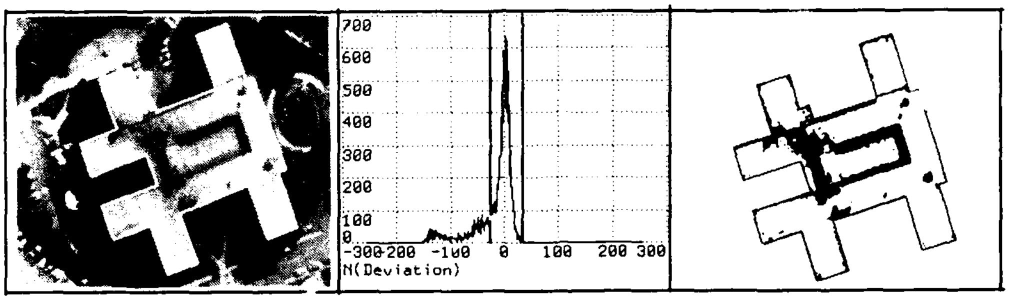
$$\begin{aligned} p(e, e_S|m) &= p(e|m)p(e_S|m, e) \\ p(e, e_S) &= p(e)p(e_S|e) \quad . \end{aligned}$$

In the case of a pair of stereo images, e is the evidence measured in the left image and e_S the corresponding evidence in the right image relative to the model projected into that image. For a stereo pair, we therefore add to the effectiveness a *stereo effectiveness* term.

$$F_S = -\log p(e_S|e) + \log p(e_S|m, e) \quad . \tag{7}$$

Area Model for Homogeneous Regions

We model the interior intensities of an image region by a smooth intensity surface with a Gaussian distribution of deviations from the surface. Since objects in real images typically have anomalies which do not lie on the smooth surface, we encode such anomalous pixels as outliers. As we shall see later, this can critically enhance the discriminatory power of the area-encoding effectiveness.



(a) (b) (c)

Figure 1: (a) Image and delineated model instance. (b) Histogram of deviations from planar fit to delineated region. (c) Pixels within indicated Gaussian peak are white; anomalous pixels outside the peak are black.

In the application of our approach to aerial imagery, we take the intensity surface to be a plane. In Figure 1, we show: (a) An image and a delineated model instance (b) The histogram of deviations from the planar fit to the intensity surface. (c) The solid white area indicating the location of the pixels within the main Gaussian peak. Black areas within the model outline lie outside the peak and are considered anomalous.

In an 8-bit image, it would take 8A bits to encode the pixel values if we did not take advantage of dependencies among pixels. Similarly, it would take $k_A A$ bits to encode the same information using our region model, where

$$k_A A = n(\log \sigma + c) + 8\bar{n} + E(n, \bar{n}) \quad . \tag{8}$$

Here $n(\log \sigma + c)$ is the cost of Huffmann encoding [12] the pixels in a Gaussian peak (see Example 1 of Appendix C), $8\bar{n}$ is the cost of encoding the outliers, and

$$E(n, \bar{n}) = \left[n \log \frac{n}{A} + \bar{n} \log \frac{\bar{n}}{A} \right] . \tag{9}$$

is the entropy, i.e., the cost of specifying whether a pixel is or is not anomalous. σ is the variance of the Gaussian distribution, n is the number of pixels in the Gaussian, and $\overline{n} = A - n$, and $c = \frac{1}{2}\log(2\pi e)$. Note that in the computation of the encoding cost, we have not included the cost of encoding the six internal parameters of the model: 3 for the plane, 2 for the Gaussian, and one for the probability n/A that a pixel lies in the main peak . It can be shown (see Appendix D) [22,25] that these costs are approximately equal to $\frac{1}{2}\log A$ bits per internal parameter of the statistical distribution, and are therefore negligibly small compared to $k_A A$.

We weight all areas and lengths using the scale parameter s (see the section "Essential Parameters of the Objective Function"), so that the area-encoding effectiveness becomes:

$$\begin{aligned} F_{I,A} &= \text{bits(area without model)} - \text{bits(area with model)} \\ &= (8 - k_A)\frac{A}{s^2} \\ &= \frac{1}{s^2}\left((8 - c - \log\sigma)n - E(n,\overline{n})\right) \quad . \end{aligned} \tag{10}$$

Optimization of this score is intuitively appropriate because it finds the best compromise among the following:

- large area A,
- low standard deviation σ,
- small number of anomalies $\overline{n}$.

Effect of Anomaly Discounting. In the graphs on the left in Figure 2, we plot the area-encoding effectiveness F_A as a function of the radius of a square patch centered at the center of the images shown in the left column: a good but noisy synthetic image of a square, the same image with edge jitter, and with gross area anomalies. When we compare the results obtained *after discounting anomalies* (solid lines) with those results found without anomaly discounting (dotted lines), we see that anomaly discounting *must* be included to make the objective function reliably select the same shape a human observer perceives. This is potentially a critical factor in the practical application of this approach because, as we see in Figure 1, real images nearly always have significant anomalous components.

Note that we only have *local* maxima of the area-encoding effectiveness appearing in Figure 1; for large radii, a better parse of the scene would be in terms of *two* model hypotheses, one square and one square-shaped ring covering the rest of the image, rather than one square plus random background. From this example, we see that high score alone is not an adequate criterion: we must also require local maximality when dealing with a partial description of the scene as opposed to a global one. For this reason it is important in practice to measure whether a candidate object passes this maximality test. We have found that this requirement is effectively enforced by requiring a minimal edge quality, and we now turn to the explicit form of the edge term used.

Edge Model

We adopt the definition [24,13,6] of edge pixels as maxima of the local image derivative, and we classify edges according to whether or not an edge boundary pixel conforms to this definition. In the absence of a model, it would take 1 bit per pixel to encode this information. If we now use the 1-parameter model that takes into account the proportion of maximal edge pixels, the most efficient Huffmann [12] code for this information would require

$$k_E = -\left[\frac{n}{L}\log\frac{n}{L} + \frac{\overline{n}}{L}\log\frac{\overline{n}}{L}\right] \tag{11}$$

bits per boundary pixel, where L is the length of patch boundary in pixels, n is the number of boundary pixels that are maxima of the local image gradient, and $\overline{n} = L - n$.

We then weight all lengths by the scale factor s and estimate the edge-encoding effectiveness to be

$$\begin{aligned} F_{I,E} &= \text{bits(edge without model)} - \text{bits(edge with model)} \\ &= (1 - k_E)\frac{L}{s} \quad . \end{aligned} \tag{12}$$

As in the case of the area term, we have neglected the $\frac{1}{2}\log\frac{L}{s}$ bits required to encode the one internal parameter of the model (see Appendix D) [22,25].

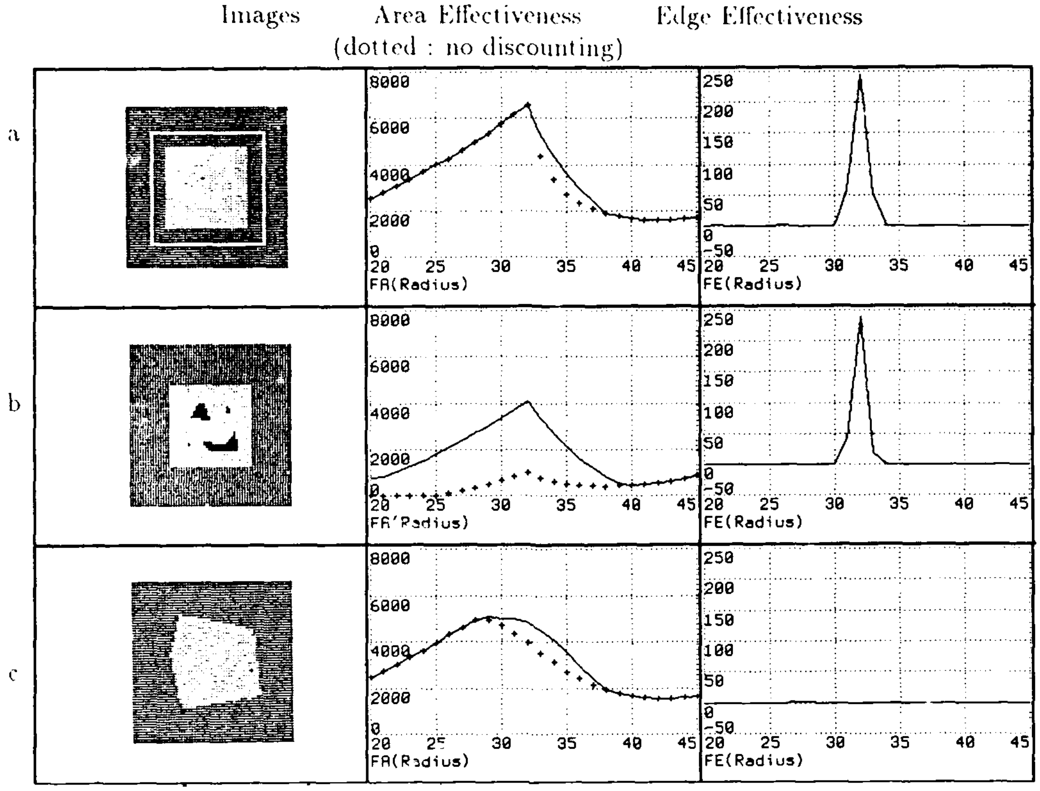


Figure 2: Area and edge effectiveness of a squared patch as a function of their radius. The patches of radius 20 and 45 are outlined on the top image.

This edge score is maximal when all boundary pixels conform to our edge model, an l degrades as the proportion of such pixels diminishes.

Stereography

The simplest stereo model assumes that corresponding pixels have the same grey-levels in both images (see, e.g., [1]). In practice, to compute the stereo effectiveness of Eq. (7), we determine the number of bits required to encode the projected patch in the second image, while knowing its photometry in the first. We compute the deviations of the intensities from their predicted values and encode them using the same Gaussian model with anomalies that we used for the area term. The anomaly discounting is required because of the possibility of occlusions. We also want to take into account the edge quality of the contour in the second image and its edge-encoding effectiveness.

We therefore take the stereographic effectiveness term F_S to be the sum of an edge and area term:

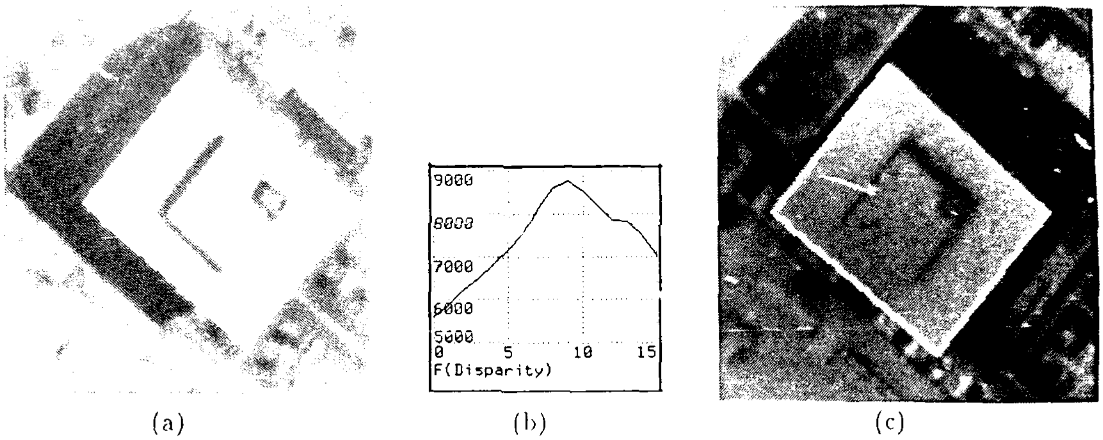
$$\begin{aligned} F_S &= F_{AS} \quad F_{ES} \\ F_{AS} &= (8 - k_{A_2})\frac{A_2}{s^2} \\ F_{ES} &= (1 - k_{E_2})\frac{L_2}{s} \end{aligned} \tag{13}$$

where A_2 is the area of the projected patch in the second image, L_2 is its boundary, and k_{A_2} and k_{E_2} are the corresponding model encoding costs.

We can use the effectiveness measure (13) to optimize the elevation parameters of a two-dimensional delineation found in the first image. The search space is extremely constrained since the projected shape is known and the only degree of freedom is epipolar motion in the second image.

Figure 3: A stereo pair of images containing a large building.

Let us consider the stereo pair of images shown in Figure 3 and the rooftop outlined in Figure 4(a). Assuming that it is horizontal, we plot in Figure 4(b) the value of F_S as a function of the assumed disparity between the outline in the left image and the outline in the right image. We note that F_S presents a sharp peak for the correct match shown in Figure 4 (c).



(a) (b) (c)

Figure 4: (a) The main rooftop in the left image of Figure 3 (b) F_S [illegible] function of the assumed disparity between left and right image. (c) The [illegible]on of the contour in the right image using the best disparity value.

Geometric Measures: Computing G.

The geometric cost G defined by Eq. (6) is a *measure of quality* of a set of object hypotheses. The simplest way to handle dependencies among objects is to require that there be no conflicts within a particular set of hypotheses; formally we write:

$$p(m_i|m_j) = p(m_i) \quad \text{if } m_i \cap m_j = \emptyset \text{ or } m_i \subseteq m_j, \quad 0 \quad \text{otherwise} \tag{14}$$

$$p(m_1 \ldots m_n) = \prod_i p(m_i) \quad \text{if no conflict}, \quad 0 \quad \text{otherwise}. \tag{15}$$

It follows that G can be expressed as

$$G = -\log p(m_1, \ldots, m_n) = \sum_{i=1}^{n} G_i. \tag{16}$$

where $G_i \propto -\log p(m_i)$ is a model quality measure that increases as the shape degrades, and γ is the arbitrary *shape coefficient*.

Now we can deduce a mechanism for deciding whether or not the addition of one more feature object is advantageous or detrimental to the overall parse. If we write the overall score in the form

$$F = \sum_{i=1}^{n}(F_i - \gamma G_i),$$

we conclude that we should accept only model instances with $(F_i - \gamma G_i) > 0$, since these are the only ones that improve the likelihood of the full scene parse.

G_i is the cost of encoding an object. In this work we take G_I to be the sum of the cost of chain-encoding the boundary of the area plus a constant cost for introducing a new object; this gives a geometric cost

$$G_i = c + \frac{L_i}{s}. \tag{17}$$

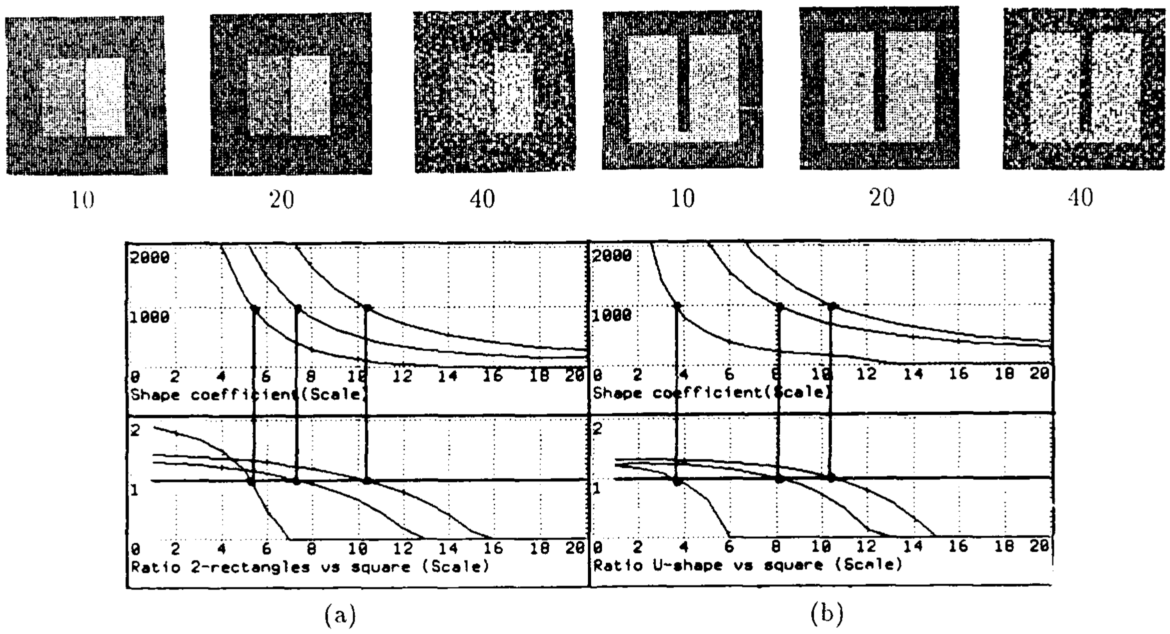


Figure 5: (a) Ratio of single-square to double-rectangle score as a function of noise variance (40, 20, 10). (b) Similar plot comparing the score of the square interpretation to the "U" interpretation.

In Figure 5(a), we show how the length term (17), which gives preference to compact objects, influences the parse when a split square is interpreted alternately as a single compact square or two adjacent rectangles. The bottom graph takes three images, with noise variance 40, 20 and 10, and plots the ratios (two-rectangle score)/(square score) as a function of scale for fixed $\gamma = 1$. Note that increasing the scale in this example amounts to looking at a reduced image in which fine details are no longer visible. The interesting point is the value of scale for which the scores are *equal*, i.e., the ratio is one. Thus we plot in the upper graphs the locus of points where the ratio is unity as a function of γ as well as scale. In Figure 5(b), we carry out a similar plot for an image of a square with a missing portion that makes it "U"-shaped. We see that the ratio ("U" score)/(square score) behaves so that the square interpretation is preferred at a large scale in the best image, and at a much lower scale in the noisier images.

Examples

We have applied the principle of objective-function optimization to operator-initiated shape extraction and to automated extraction of generic cartographic features such as buildings from aerial imagery, both described elsewhere in these Proceedings [10]. In the automated application, we use an hypothesis generator that carries out the following steps: (1) extract linked edges; (2) find edges obeying geometric constraints (such as rectilinearity) that define enclosed regions in the image; (3) compute the score of each enclosed area using the objective function; (4) find the subset of nonconflicting shape candidates maximizing the total score.

The objective function plays a crucial role in this application because the hypothesis generator will always produce conflicting sets of candidates, and a means of distinguishing among these is absolutely essential. In the remainder of this section, we show how the output of this system depends on the choice of the scale parameter for a fixed shape coefficient.

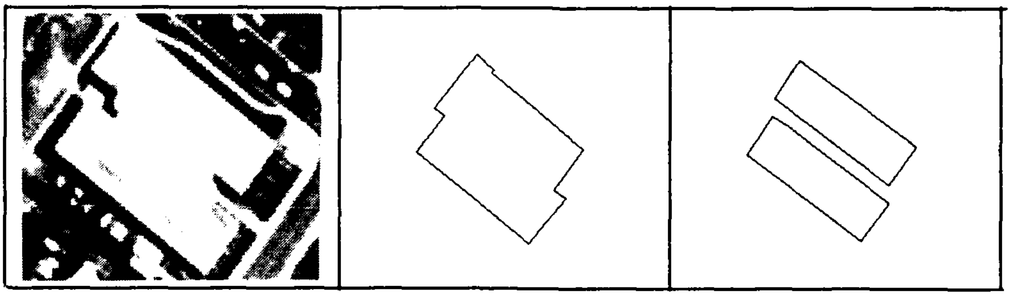

(a) (b) (c)

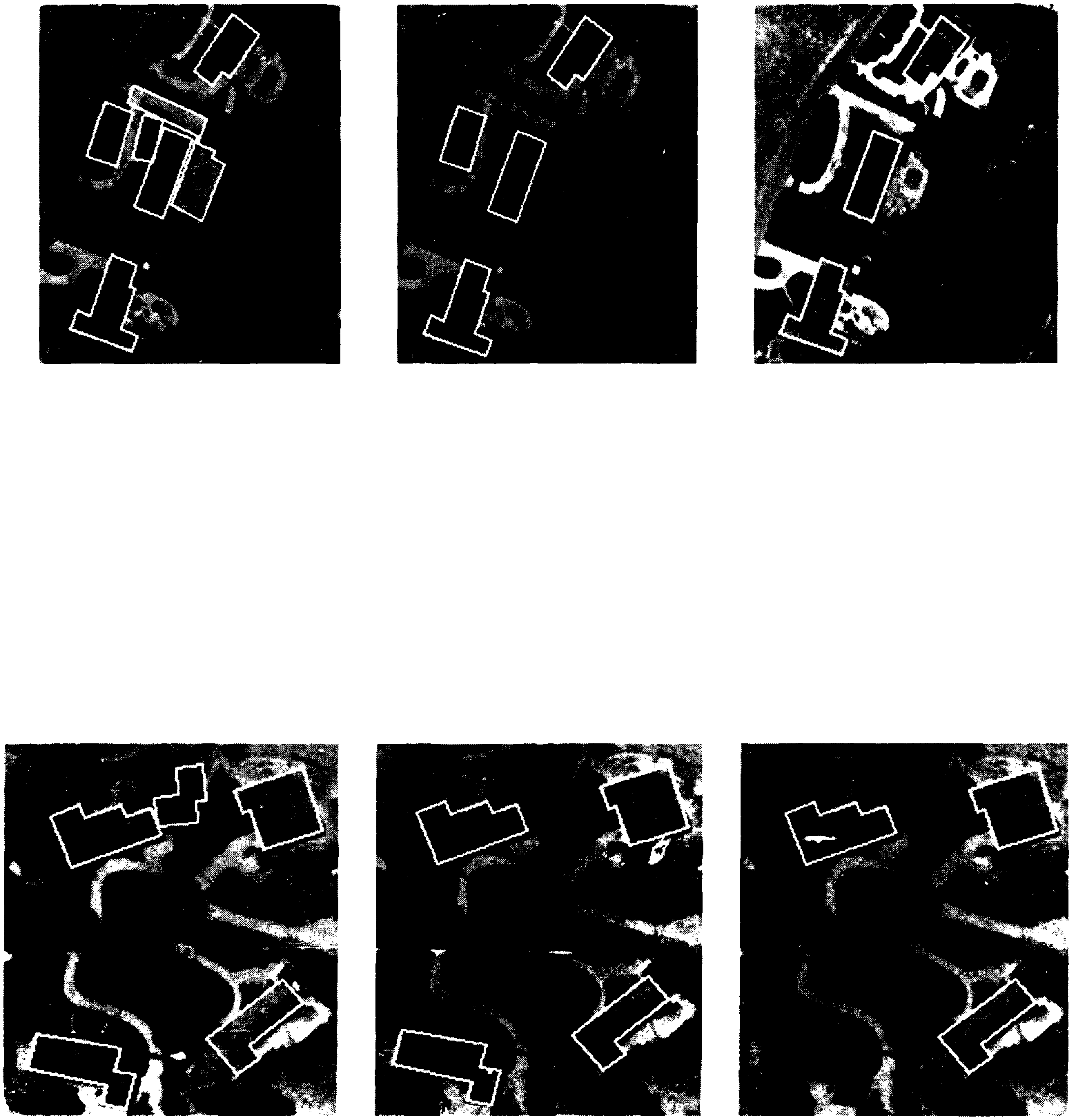
Figure 6: (a) A complex building. (b) Interpretation in terms of a single polygon. (c) Interpretation in terms of two polygons.

The automated system produces two conflicting interpretations of the building in Figure 6(a): one in terms of a single polygon enclosing both wings as in Figure 6(b) the other in terms of two polygons, one for each wing as in Figure 6(c). At low scale the latter will be preferred because of its better fit to the photometric data, while at high scale the former will dominate due to its lower geometric cost. In the top row of Figure 7, we present the best subset of candidate object models selected for various values of the scale. Note that at high scale, only objects that conform best to the model are retained, regardless of their size. Conversely, at low scale, objects that only marginally conform to the model are retained. In the bottom row of Figure 7, we show a similar example in which hypotheses generated by the system are ranked by the objective function as a function of the scale. The scale effectively acts as a quality filter, determining the minimal characteristics of the features one wishes to accept.

From the examples shown in this Section, combined with the knowledge in Appendix C, we can begin to understand the important qualitative properties of the scale parameter: s tunes the scale *not* of the physical size of the object, but the scale of its *quality*. Objects with close fits to the strict model are selected first as we ramp the scale down from a high value.

Conclusion

In this work, we have shown how an information theoretic approach to the feature extraction problem can be formulated in such a way as to permit realistic computational techniques for the required probability estimates. Our approach provides a firm theoretical basis for understanding complex feature extraction problems that require a balance between photometric evidence and geometric quality. Of course, the objective function approach given here cannot by itself lead to good solutions to the feature extraction problem, but must be teamed with a competent (human or automated) hypothesis generator. Applications of the objective function approach to an interactive system and to an automated hypothesis generator for extracting buildings from aerial imagery are described in [10]. Among the goals of future work will be the extension of the range of our models and the treatment of complex semantic dependencies in terms of their information-theoretic context.



Scale 6 Scale 7 Scale 8

Figure 7: Top row: an aerial image of suburban buildings parsed at scales 6, 7, and 8.
Bottom row: a similar image at scales 6, 7, and 8.

Appendix A: Derivation of the Objective Function

In this app :ndix, we prove Eq. (1) given the assumptions stated in the text.

We reiterate the definition of P as the probability that, given the evidence $E = \{e_i;\ i = 1 \ldots n\}$, parsing the scene in terms of a particular set of model instances $M = \{m_i;\ i = 1 \ldots n\}$ is in fact correct.

Our assumptions can be stated mathematically as follows: Let I, J, K denote sets of indices referring to model instances and their corresponding bodies of evidence. Then we can assume the following:

Assumption 1: Evidence has no bearing on any proposition that does not include knowledge of the geometric location of the evidence. That is, when $J \cap I = \emptyset$ and $J \cap K = \emptyset$,

$$P(m_J e_K | e_I) = P(m_J e_K).$$

Assumption 2: Conditioning on a model proposition in conjunction with the evidence supporting it is the same as conditioning on the model proposition alone; the evidence is irrelevant, since the conditional probability already presumes complete validity of the proposition m_I being conditioned upon. Thus for any J,

$$P(m_J | m_I, e_I) = P(m_J | m_I).$$

From the:e assumptions and Bayes' theorem, we prove the following:

Corollary 1: When $I \cap J = \emptyset$, for any set of propositions X with no indices in common with J,

$$\begin{aligned} P(m_I | X, e_J) &= \frac{P(m_I, X | e_J)}{P(X | e_J)} \\ &= \frac{P(m_I, X)}{P(X)} \\ &= P(m_I | X) \end{aligned} \tag{18}$$

where the second line follows from Assumption 1.

Corollary 2: For any set of propositions X with no indices in common with J,

$$\begin{aligned} P(m_I | X, m_J, e_J) &= \frac{P(m_I, X | m_J, e_J)}{P(X | m_J, e_J)} \\ &= \frac{P(m_I, X | m_J)}{P(X | m_J)} \\ &= P(m_I | X, m_J) \end{aligned} \tag{19}$$

where the second line follows from Assumption 2.

Corollary 3: For any set of propositions X with no indices in common with I,

$$\begin{aligned} p(X, e_I) &= p(X | e_I) p(e_I) \\ &= p(X) p(e_I), \end{aligned} \tag{20}$$

where the second line follows from Assumption 1.

Corollary 4: For any set of propositions X with no indices in common with I,

$$\begin{aligned} p(X, e_I|m_I) &= p(X|m_I, e_I)p(e_I|m_I) \\ &= p(X|m_I)p(e_I|m_I) \end{aligned} \tag{21}$$

where the second line follows from Assumption 2.

The proof of Eq. (1) thus proceeds from applying the conditional probability decomposition formula $p(a, b|c) = p(a|b, c)p(b|c)$, and the corollaries listed:

$$\begin{aligned} P &= p(m_0, m_1, \ldots, m_n|e_1, \ldots, e_n) \\ &= \textstyle\prod_i p(m_i|m_1, \ldots, m_{i-1}, e_1, \ldots, e_n) && (RecursiveDecomposition) \\ &= \textstyle\prod_i p(m_i|m_1, \ldots, m_{i-1}, e_1, \ldots, e_{i-1}, e_i) && (Corollary1) \\ &= \textstyle\prod_i p(m_i|m_1, \ldots, m_{i-1}, e_i) && (Corollary2) \\ &= \textstyle\prod_i \frac{p(m_1,\ldots,m_{i-1},e_i|m_i)p(m_i)}{p(m_1,\ldots,m_{i-1},e_i)} && (Bayes' Rule) \\ &= \textstyle\prod_i \frac{p(m_1,\ldots,m_{i-1},e_i|m_i)p(m_i)}{p(m_1,\ldots,m_{i-1})p(e_i)} && (Corollary3) \\ &= \textstyle\prod_i \frac{p(m_1,\ldots,m_{i-1}|m_i)p(e_i|m_i)p(m_i)}{p(m_1,\ldots,m_{i-1})p(e_i)} && (Corollary4). \end{aligned}$$

The final step is to use Bayes' rule again to regroup the terms involving only m_j, yielding

$$\begin{aligned} P &= \prod_i p(m_i|m_1, \ldots, m_{i-1}) \prod_{i>0} \frac{p(e_i|m_i)}{p(e_i)} \\ &= p(m_0, m_1, \ldots, m_n) \prod_{i>0} \frac{p(e_i|m_i)}{p(e_i)}, \end{aligned}$$

which is our final result.

Appendix B: Validity of the Independence Assumptions

In this appendix, we discuss the implications of the independence assumptions stated in Appendix A.

As before, let I, J, K denote sets of indices referring to model instances and their corresponding bodies of evidence.

Corollary 5: We prove that

$$p(e_I|m_I, m_J) = p(e_I|m_I)$$

using the following argument:

When $I \cap J = \emptyset$,

$$\begin{aligned} p(m_I, e_I|m_J) &= \frac{p(m_J|m_I, e_I)}{p(m_J)} p(m_I, e_I) \\ &= \frac{p(m_J|m_I)}{p(m_J)} p(m_I, e_I) \\ &= \frac{p(m_J|m_I)}{p(m_J)} p(m_I)p(e_I|m_I) \\ &= p(m_I|m_J)p(e_I|m_I) \end{aligned}$$

where the second line follows from Assumption 2, Appendix A. By Bayes' rule,

$$p(e_I|m_I, m_J) = \frac{p(e_i, m_I|m_J)}{p(m_I|m_J)}.$$

We interpret this result to mean that, when our assumptions are valid, the photometry of an object is not influenced by other objects in the image. Our assumptions are therefore reasonable in images with moderate feature density to the extent that features are not occluded or shadowed by other features.

Appendix C: Computing Encoding Costs

Closely following Leclerc [14,15, pages 367–369] we define the problem of computing encoding costs of a body of data in terms of an encoder, whose input is the set of observations in the form of a bit string, and whose output is supplied to a decoder, which uses this to produce the output.

When the output of the decoder is identical to the input of the encoder, then the output of the encoder is called an information preserving description of the input, denoted $\mathcal{D}_i^j$(input), written in the *descriptive language* L_i of the decoder. The superscript j indicates that there may, in general, be more than one possible description for a given input. The difference between the number of bits in a trivial description and the description in terms of L_i is what we define in this paper as the *effectiveness* of the description.

The most efficient description, $\mathcal{D}_i^*$(input), is the minimal-length information-preserving description of the input written in the descriptive language of the decoder.

Minimal-Length Descriptions of Ergodic Processes

Consider the case when the input string is generated by a known ergodic process $\mathcal{F}$, that is,

$$\text{input} = F(\mathbf{X}),$$

where $\mathbf{X} = \{x_0, x_1, \ldots, x_{n-1}\}$ is drawn from a known unchanging distribution; the length of the vector, n, may also a be random variable. P(input) is therefore defined for every input string and the pair $(F, \mathbf{X})$ constitutes a description of the input. When the function is not uniquely invertible, i.e., when a given input string can be produced by many different $\mathbf{X}$'s, the problem of how to choose a single $\mathbf{X}$ for a given input arises.

The minimal-length description solution to the problem of choosing a single description is based on the observation that it is always possible to design an optimal descriptive language $L_{\mathcal{F}}$ for an ergodic process $\mathcal{F}$ such that the shortest description of the input has the length

$$|\mathcal{D}_{\mathcal{F}}^*(\text{input})| = -\log_2 P(\text{input}), \tag{22}$$

in bits[3] [12]. Such a descriptive language is optimal in the sense that no other descriptive language can expect to produce a shorter description than this, on the average.[4] A consequence of this optimality is that there exists a unique shortest description for every input string,[5] because otherwise there would exist "wasted" descriptions, those that map to the same input, that could have been used for other inputs but were not; hence, one could have devised a more efficient descriptive language that made use of these "wasted" descriptions. Note, however, that there are always many different optimal descriptive languages for a given ergodic process, but they are equivalent to each other in the sense that there exist one-to-one mappings between them, as a consequence of the uniqueness of the description.

Example 1: Independent Symbols

The purpose of this example is to illustrate the design of an optimal descriptive language for input strings consisting of symbols independently drawn from a known distribution. In this example, the input string is written $\{x_0, x_1, \ldots, x_n\}$, where x_i is one of the three symbols in the set $X = \{a, b, c\}$, independently drawn from a distribution such that a occurs with probability 0.5, and b and c each occur with probability 0.25. They are encoded in a straightforward fashion for input to the encoder as, say, a ='00,' b ='01,' and c ='10;' thus a string of n symbols will require $2n$ bits as input. From Eq. (22) above, we can design a descriptive language L_1 such that the description length is

$$\begin{aligned}
|\mathcal{D}_1(\text{input})| &= -\log_2 P(\{x_0, x_1, \ldots, x_n\}) \\
&= -\log_2 \prod_{i=0}^{n} P(x_i) \\
&= \sum_{i=0}^{n} -\log_2 P(x_i).
\end{aligned}$$

[3] For some distributions, one would need to encode an infinitely long input string in order to achieve exactly this efficiency. A more precise statement is that we can achieve an efficiency as close to this optimum as we like by encoding sufficiently large chunks of the input string at a time.

[4] This is not to say that no other descriptive language can do better on any given finite input string, but only that no other language can do better on the average, or, equivalently, no other language can do better for arbitrarily long input strings.

[5] The phrase "for every input string" used here and elsewhere is short for "for every input string that has non-zero probability of occurrence."

Since $-\log_2 P(a) = 1$ and $-\log_2 P(b) = -\log_2 P(c) = 2$, a code devoting exactly one bit for symbol a and two bits each for b and c, such as the *instantaneous* code a ='0,' b ='10,' and c ='11,' is an optimal code. So, for example, the 16-symbol input string *aabccbaabacabcaa* would be encoded as '001011111000100110101100,' requiring 24 bits (1.5 bits per symbol) instead of the 32 bits that a fixed-length input code (2 bits per symbol) would require. In general, $-\log_2 P(x_i)$ is not an integer, so we must encode several symbols at once (called block encoding) in order to achieve something close to the optimal encoding length; however, the principle remains the same.

In the computation of edge and area encoding effectivenesses of Eq. (12) and Eq. (10), we directly use Eq. (23) to estimate the cost of encoding whether or not boundary pixels satisfy our edge criterion and whether or not interior pixels are outliers. Similarly, we model the deviations from the planar fit of the interior area photometry by a normal distribution $\mathcal{N}(0, \sigma^2)$. These deviations are rounded and represented within the histogram with a precision of 1. Assuming that they are drawn from the normal distibution, the probability of an element with deviation r is then

$$\begin{aligned} P(r) &= \int_{\lfloor r \rfloor_1}^{\lceil r \rceil_1} \frac{1}{\sqrt{2\pi\sigma^2}} \exp\left[\frac{-x^2}{2\sigma^2}\right] dx \\ &\approx \frac{1}{\sqrt{2\pi\sigma^2}} \exp\left[\frac{-r^2}{2\sigma^2}\right] \quad . \end{aligned}$$

assuming that $\sigma > 1$. The total cost of encoding the n pixels within the gaussian peak then is:

$$\begin{aligned} C &= \sum -\log \frac{1}{\sqrt{2\pi\sigma^2}} \exp\left[\frac{-r^2}{2\sigma^2}\right] \\ &= n\frac{\log 2\pi}{2} + n \log e \left(\log_e \sigma + \frac{1}{2\sigma^2}\frac{\sum r^2}{n}\right) \end{aligned}$$

It is easy to see that C is minimized when σ is equal to the variance $\sum r^2/n$ of the deviations, and is given by:

$$C = n(\frac{1}{2}\log(2\pi e) + \log\sigma)$$

Although the first order statistics of an input string (the probability of occurrence of a symbol) captures some aspect of the structure of the input, and indeed, captures it entirely when the symbols are independent and identically distributed as in this example, the next example illustrates that much more is needed.

Example 2: Correlated Symbols

Consider a similar case to the one above where the three symbols a, b, and c also occur with probability 0.5, 0.25, and 0.25, but such that they only occur as sequences *aabc* or *bcaa*, with equal probability. Clearly, we need only 1 bit to distinguish one sequence from the other, so that an optimal descriptive language L_2 would be: *aabc*='0' and *bcaa*='1.' So, for example, the input string *aabccbaacbaaaabc* would be encoded as '0110,' requiring 4 bits (0.25 bits per symbol) instead of the 24 bits required for $\mathcal{D}_1$ (1.5 bits per symbol) or the 32 bits (2 bits per symbol) used as input to the encoder.

The above is an example of the more general case of a Markov Random Process (MRP), where $P(x_i)$ depends on $\{x_{i-1}, x_{i-2}, \ldots, x_{i-m}\}$, but is independent of the other input symbols. The optimal descriptive language for a MRP is no longer a straightforward mapping of each symbol (or group of symbols) to a unique bit string. The technical details are not important here, but the basic idea is that the coding scheme for x_i becomes a function of $\{x_{i-1}, x_{i-2}, \ldots, x_{i-m}\}$. Nonetheless, the code is still unique and the number of bits required to encode x_i is still $-\log_2 P(x_i)$, but now $P(x_i)$ depends on $\{x_{i-1}, x_{i-2}, \ldots, x_{i-m}\}$ instead of being fixed *a priori*. Thus, one encodes the first m symbols in the straightforward fashion of Example 1 to start things off, then one encodes the next symbol using the code defined by these m symbols, then the next symbol using the code defined by the previous m symbols, and so on. Thus, the decoder must know the encoding scheme for, or, equivalently, the probability distribution of, each possible combination of m symbols. This makes the decoder more complex, of course, but allows for the optimal encoding of MRP's.

Here we see that higher order statistics (the conditional probability of occurrence of an input symbol, given a subset of the others) allowed us to capture more of the structure of the input than simple first order statistics, at least for MRP's.

Example 3: Encoding images using a Laplacian Pyramid

As noted in the previous example, one can exploit dependencies to efficiently encode data. In the case of images, Burt et al. [5] have proposed a data compression scheme that achieves that effect. To encode an image, pixel-to-pixel correlations are first removed by subtracting a low-pass filtered copy of the image from the image itself. The net result is that the data is compressed, since the difference, or error, image has low variance and entropy, and the low-pass filtered image may be represented at reduced sample density. These steps are then repeated to recursively compress and reduce each low-pass image. *Iteration of the process generates a pyramid data structure.*

In this pyramid, the upper levels, which are very cheap to encode, describe the low frequencies, while the lower levels, which are more expensive, represent the high frequencies. Using this scheme it takes about 2 bits per pixel to completely encode an 8-bit image with large-scale structures typical of aerial images. We can also reconstruct an image using only the upper levels of the pyramid while ignoring the lower ones. The resulting image can be encoded using far less than 2 bits per pixel, but lacks the high frequencies of the original image and appears to be a blurred version of it. However, if we are only interested in large scale structures in the image, the blurred image may have all the information we need. Consider once again the example of a shingled roof. The general shape of the roof may be adequately described in the low frequency image while the shingles are not, since they correspond to higher frequencies. In a system like ours, we are interested in the global description of the roof but not of the individual shingles: the information encoded in the low frequency image is therefore perfectly adequate for our purpose.

In this context, we can better understand the role of the *scale parameter* s introduced in the main text. In the absence of a model, the information in an 8-bit image can be encoded using $8/s^2$ bits per pixel. Increasing s amounts to describing the image using fewer and fewer bits of information, which, in the pyramid encoding scheme, can be done by omitting the lower levels of the pyramid and ignoring the high frequencies in the image. The scale can therefore be regarded as a measure of the maximal (Nyquist) frequency of interest in the image, the higher frequencies being regarded as irrelevant noise. The relevant data in the image can be faithfully represented by sampling the signal at twice this frequency.

Appendix D: Internal Parameter Encoding Cost

Each model can have an arbitrary set of internal parameters $\{\theta\}$, such as the three parameters needed to specify a plane, so that

$$p(\epsilon_i|m_i) = \int d\theta \;\; p(\epsilon_i|m_i,\theta).$$

However, as shown in references [22,25], $\log p(e_i|m_i)$ [illegible] e estimated by finding the optimal θ and using

$$\begin{aligned}\log p(\epsilon_i|m_i) &= \log \int d\theta \;\; p(\epsilon_i|m_i,\theta) \\ &\approx \max_{\theta} \log p(\epsilon_i|m_i,\theta) - \frac{k}{2}\log N,\end{aligned} \tag{23}$$

where k is the number of parameters in $\{\theta\}$ and N is the total number of data samples used to evaluate this model. Thus, in our objective functions, we need not explicitly deal with the internal parameters $\{\theta\}$; in fact, the logarithmic contribution is normally so small relative to the other terms that we can omit this term in practice. For further details, we refer the reader to the original literature.

References

[1] S. Barnard, "Stochastic Stereo Matching Over Scale," Proceedings of the Image Understanding Workshop, pp. 769 778 (April 1988).

[2] T.O. Binford, "Survey of Model-Based Image Analysis Systems," The International Journal of Robotics Research 1, No. 1, pp. 18 64 (Spring, 1982).

[3] R.C. Bolles and R. Horaud, "3DPO, A Three-Dimensional Part Orientation System," International Journal of Robotics Research 5, pp. 3 26 (1986).

[4] R.A. Brooks, "Symbolic Reasoning Among 3-D Models and 2-D Images," Artificial Intelligence Journal **16**, (1981).

[5] P.J. Burt and E.H. Adelson, "The Laplacian Pyramid as a Compact Image Code," IEEE Trans. on Comm. **4**, pp. 532–540 (1983).

[6] J. Canny, "A Computational Approach to Edge Detection," IEEE Trans. PAMI **8**, pp. 679–698 (1986).

[7] J.A. Feldman and Y. Yakimovsky, "Decision Theory and Artificial Intelligence: I. A Semantics-Based Region Analyzer," Artificial Intelligence **5**, pp., 349–371 (1974).

[8] M.A. Fischler, J.M. Tenenbaum, and H.C. Wolf, "Detection of Roads and Linear Structures in Low-Resolution Aerial Imagery Using a Multisource Knowledge Integration Technique," Computer Graphics and Image Processing **15**, pp. 201–223 (1981).

[9] P. Fua and A.J. Hanson, "Extracting Generic Shapes Using Model-Driven Optimization," Proc. Image Understanding Workshop, Boston, MA, pp. 994–1004 (April 1988).

[10] P. Fua and A.J. Hanson, "Objective Functions for Feature Discrimination: Applications to Semiautomated and Automated Feature Extraction," in this Proceedings, Proceedings of the Image Understanding Workshop, Palo Alto, CA (May 1989).

[11] M.P. Georgeff and C.S. Wallace, "A General Selection Criterion for Inductive Inference," in Proceedings of Advances in Artificial Intelligence, Pisa , Italy Sept. 1984, T. O'Shea (Ed.), North Holland, Amsterdam (1984).

[12] R.W. Hamming, "Coding and Information Theory," Prentice Hall, New Jersey (1985).

[13] R.M. Haralick, "Digital Step Edges from Zero Crossings of Second Directional Derivatives," IEEE Trans. PAMI **6**, pp. 58–68 (1984).

[14] Y.G. Leclerc, private communication.

[15] Y.G. Leclerc, "Image Partitioning as Constructing *Simple Stable Descriptions*," *Proc. Image Understanding* Workshop, Boston, MA (April 1988); International Journal of Computer Vision, *in press.*

[16] Y.G. Leclerc and P. Fua, "Finding Object Boundaries Using Guided Gradient Ascent," Proceedings of the the 1987 Topical Meeting on Machine Vision, Lake Tahoe, California, pp. 168–171 (March 1987). Also presented at the DARPA Image Understanding Workshop, Los Angeles, California, pp. 888–891 (February 1987).

[17] P. Fua and Y.G. Leclerc, "Model Driven Edge Detection," Machine Vision and Applications, *in press.*

[18] D.M. McKeown and J.L. Denlinger, "Map-guided feature extraction from aerial imagery," Proc. 2nd IEEE Workshop on Computer Vision, pp. 205–213 (May, 1984).

[19] A. Huertas and R. Nevatia, "Detecting Buildings in Aerial Imagery," Computer Vision, Graphics and Image Processing **41**, pp. 131–152 (1988).

[20] Y. Ohta, T. Kanade, and T. Sakai, "A Production System for Region Analysis," Proc. 6th Inter. Joint Conf. on Artif. Intell., pp. 684-686 (1979).

[21] L.H. Quam, "Road Tracking and Anomaly Detection in Aerial Imagery," Proceedings: Image Understanding Workshop, pp. 51–55 (May 1978).

[22] J. Rissanen, "A Universal Prior for Integers and Estimation by Minimum Description Length," The Annals of Statistics **2**, pp. 416–431 (1983).

[23] J. Rissanen, "Minimum-Description-Length Principle," in Encyclopedia of Statistical Sciences, **5**, pp. 523-527 (1987).

[24] A. Rosenfeld, " A Nonlinear Edge Detection Technique," Proc. IEEE **58**, pp. 814–816 (1970).

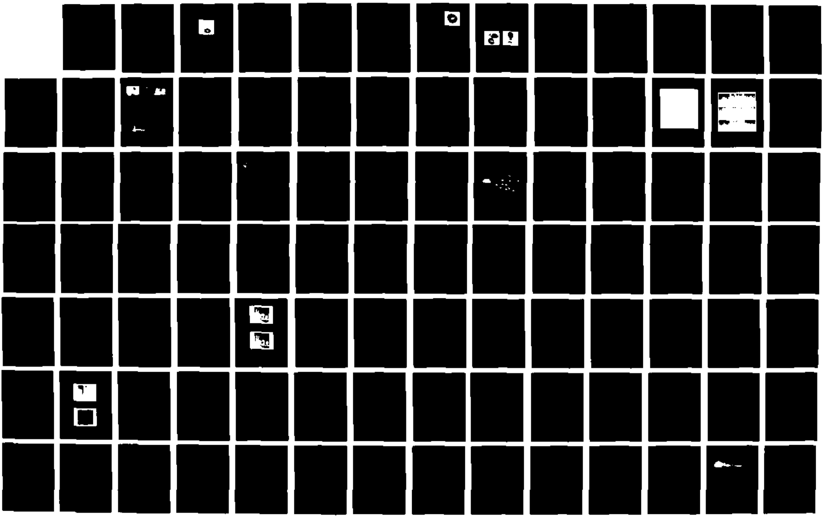
[25] G. Schwarz, "Estimating the Dimension of a Model," The Annals of Statistics **6**, pp. 461–464 (1978).

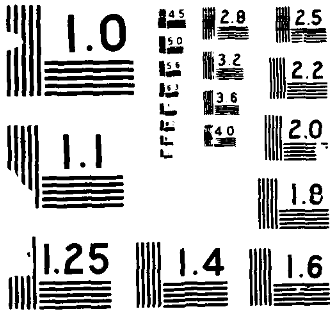
[26] C.E. Shannon, "A Mathematical Theory of Communication," Bells Systems Tech J., **27**, pp. 623–656 (1948).

[27] M.O. Shneier, R. Lumia, and F.W. Kent, "Model-Based Strategies for High-Level Robot Vision," *Computer Vision, Graphics and Image Processing* **33**, pp. 293–306 (1986).

[28] D. Terzopoulos, "On Matching Deformable Models to Images," Topical Meeting on Machine Vision, Technical Digest Series, Optical Society of America, Washington, D.C., (12):160–167 (1987).

AD-A212 981 PROCEEDINGS OF IMAGE UNDERSTANDING WORKSHOP HELD IN 6/12
PALO ALTO CALIFORNIA O. (U) SCIENCE APPLICATIONS
INTERNATIONAL CORP MCLEAN VA L S BAUMAN MAY 89
UNCLASSIFIED N00014-86-C-0700 F/G 17/11 NL




1.0
4.5
5.0
5.6
6.3
2.8
3.2
3.6
4.0
2.5
2.2
2.0
1.1
1.8
1.25
1.4
1.6

ON RECOGNIZING AND POSITIONING CURVED 3D OBJECTS FROM IMAGE CONTOURS

Jean Ponce and David J. Kriegman

Robotics Laboratory
Department of Computer Science
Stanford University, CA 94305

ABSTRACT

A new approach is presented for explicitly relating the shape of image contours to models of curved three-dimensional objects. This relationship is used for object recognition and positioning. Object models consist of collections of algebraic surface patches and their intersection curves; this includes nearly all representations used in computer aided design and computer vision. The image contours considered are the projections of surface discontinuities and occluding contours. Elimination theory provides a method for constructing the implicit equation of the image contours of an object observed under orthographic or perspective projection. This equation is parameterized by the object's position and orientation with respect to the observer. Determining these parameters is reduced to a fitting problem between the theoretical contour and the observed data points. The proposed approach readily extends to parameterized models. It has been implemented for a simple world composed of tori of different sizes and successfully tested on several real images. Other new applications of elimination theory to computer vision and graphics are briefly discussed.

1. INTRODUCTION

This paper addresses the recognition and positioning of curved three-dimensional objects from their monocular image contours under the following assumptions: Precise geometric models of the observed objects (and/or object classes) are available. The data consists of imperfect edge-maps; in particular, knowledge of high level features such as junctions or corners is not required. No additional information such as shading, surface normals, or range is available.

Solving this problem is important for several reasons: The world around us is *not* piecewise planar, but composed of curved objects. People can, after all, recognize familiar (and not so familiar) objects from line drawings, and intelligent robots should be able to mimic this ability. From a more pragmatic point of view, databases of CAD models are now available for a wide range of manufactured objects; opportunities are better than ever for computer vision systems to exploit these models in industrial environments. Furthermore, recognizing three-dimensional curved objects from their contours remains one of the most challenging problems in computer vision, and, as noted by Besl: "No one has yet demonstrated a solution to the occluding curve to surface (or volume) model matching problem for arbitrary objects with smooth surfaces.." [4].

Why is this so difficult? Solving this problem involves comparing the shape of the two-dimensional surfaces that bound the observed objects to the one-dimensional curves that form their image contours. For polyhedra, this is relatively easy, since the contour generators (edges) of these objects are view-independent. Indeed, some success in recognizing and positioning polyhedra has been achieved, not only from range data [20,23], but also from image contours [26,28,36,52]. The situation is quite different for curved objects, whose contour generators move and deform over the surface according to the observer's position. Very little success has been obtained, even when additional information, such as range, is available [7,18,41]. Acronym [9] probably remains the only working vision system to have successfully recognized three-dimensional curved objects from their image contours. Despite its achievements, even Acronym has severe limitations, such as a limited scope (the primitives are essentially cylinders and cones), and a limited range of admissible viewing directions (essentially overhead views). Other model-based approaches rely on view-independent features, such as vertices, or global shape descriptors such as moments. See [5] for a review of many of these approaches.

To make real progress, it is necessary to understand the geometry of image contours and to explicitly relate their shape to the shape of the observed objects and to the viewing parameters. Most of the related research is in the area of line-drawing interpretation and shape from contour. Line-drawing interpretation algorithms attempt to label image curves as different types of occluding contours and surface discontinuities. Although rigorous schemes have been developed for labelling the line-drawings of objects with planar [14,27,52] and even curved [38] faces, these approaches rely on perfect segmentations and are too brittle to cope with real-world images. Shape from contour approaches attempt to determine constraints on three-dimensional scene structure based on different assumptions about shape. Very few general results are available [32], and shape from contour methods usually rely on heuristics, such as smoothness [3] or compactness [8] measures, or are only applicable for certain object shapes such as planar [31,43] or curved [51,53] skewed symmetries, solids of revolution [40], or generalized cylinders [39,44,45,50].

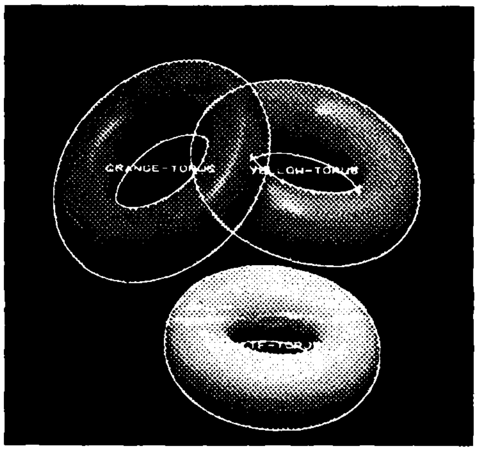


Figure 1. A real image of three tori. The recognized models are overlaid in the 3D position and orientation found by the algorithm described in section 5. The data used for recognition consists of an imperfect edge-map.

This paper proposes a new approach for explicitly relating the shape of image contours to models of curved three-dimensional objects. Object models consist of collections of algebraic surface patches and their intersection curves; this includes nearly all representations used in computer aided design and computer vision, such as CSG models, generalized cylinders, and superquadrics [35]. The image contours considered are the projections of surface discontinuities and occluding contours. Elimination theory [16,22,37,48,49] provides a method for constructing the implicit equation of the image contours of an object observed under orthographic, weak perspective, or perspective projection. This equation is parameterized by the position and orientation of the object with respect to the observer. Determining these parameters is reduced to a fitting problem between the theoretical contour and the observed data points. Two measures of fit are proposed: The implicit equation can be directly fitted to the data points. Alternatively, elimination theory can be used to construct a closed-form expression for the distance between an image point and the theoretical contour. Position and orientation are then determined by minimizing the average distance over the data points. The proposed approach readily extends to parameterized models, whose contour equation simply includes additional shape parameters. A simple recognition and positioning system has been implemented for a world composed of tori of different sizes and flavors, and it has been successfully tested on several real images of objects such as plastic rings, doughnuts, and bagels (figures 1,3,4).

2. ELIMINATION, PARAMETRIC PATCHES, AND IMPLICITIZATION

Elimination theory [16,22,37,48,49] is a classical branch of mathematics which has been "rediscovered" recently in the context of computer graphics [30], computer aided design [22,49], robot inverse kinematics [10], and robot motion planning [12]. Curiously, it does not seem to have been applied to computer vision yet, although related algebraic approaches have been used for recognition of polyhedra [15,52]. In this section, we briefly introduce elimination theory, then define parametric surface patches, and, following [22,49], show how to construct their implicit equations and represent their intersection curves using elimination.

The basic idea of elimination theory is that it is possible to express a necessary and sufficient condition for a system of algebraic equations to have common roots as the vanishing of a single polynomial called their resultant. The original variables are eliminated, and do not appear in the resultant, which is a polynomial in the coefficients of the original equations. Sylvester resultants [48] can be readily used to eliminate one variable between two polynomials. Elimination of $n-1$ variables between n polynomials can be achieved by eliminating these variables one by one with Sylvester resultants. Other elimination methods are available, and are sometimes more efficient: Bezout and Caylay resultants [16,48], Gröbner bases [10], multivariate resultants and resolvants [12,37]. In this paper, we mostly consider elimination theory as a "black box", and suppose that it is always possible to eliminate any number of variables between given equations. Limitations of the method and efficiency considerations are briefly discussed in the conclusion.

In the rest of the paper, objects are modelled by collections of parametric patches and their intersection curves. A parametric patch in standard position is defined by rational splines, i.e.,

$$\mathbf{x}(s,t) = \frac{1}{\sum_{i,j} d_{ij} s^i t^j} \sum_{i,j} s^i t^j \mathbf{x}_{ij}, \quad (s,t) \in I \times J, \tag{1}$$

where the d_{ij} are scalar coefficients, and the $\mathbf{x}_{ij}$ are vectors of coefficients. Bicubic patches are the most prominent type of surfaces in computer aided geometric design [19,49], and are given by the above equation with a maximum degree of 3 for s and t. As suggested in [22,49], it is possible to compute an implicit algebraic equation for any parametric rational patch. By multiplying the coordinates (x, y, z) of $\mathbf{x}$ by their common denominator, a system of three polynomial equations in s and t is obtained:

$$\sum_{i,j}(d_{ij}x - x_{ij})s^i t^j = 0, \quad \sum_{i,j}(d_{ij}y - y_{ij})s^i t^j = 0, \quad \sum_{i,j}(d_{ij}z - z_{ij})s^i t^j = 0, \tag{2}$$

where (x_{ij}, y_{ij}, z_{ij}) are the coordinates of $\mathbf{x}_{ij}$. By eliminating s and t between these three equations, a single polynomial equation in x, y, and z is obtained:

$$\sum_{i,j,k} a_{ijk} x^i y^j z^k = 0, \tag{3}$$

which is the implicit equation of the parametric patch. Moreover, if $\mathbf{y}(u,v)$ is another parametric patch, then the intersection curves of the two patches can be obtained by computing the implicit equation of $\mathbf{y}$, substituting the parametric coordinates of $\mathbf{x}$ in this equation, and multiplying by the denominators. The intersection curves are then given by:

$$\sum_{i,j} b_{ij} s^i t^j = 0 \tag{4}$$

which is an implicit equation of these curves in the parameter space of $\mathbf{x}$; b_{ij} is a polynomial in the coefficients of $\mathbf{x}$ and $\mathbf{y}$.

3. CONTOUR EQUATION

Image contours are generated from first order discontinuities in intensity. In turn these discontinuities are generically formed by an object surface normal discontinuity (a crease, edge or corner), depth discontinuity (occluding contours or limbs), reflectivity discontinuity (pigmentation or material changes), and lighting discontinuities (shadows) [6]. Below we show how to obtain implicit equations for the image contours formed by the projections of edges and occluding contours.

3.1 PROJECTION GEOMETRY

Different projection models are possible: orthography, weak perspective, perspective. Pure orthography is not very useful without a priori knowledge of the position of the camera, so we concentrate on weak perspective and perspective in the rest of the paper.

Let $\mathbf{R}$ be the rotation that maps the camera coordinate frame onto the world coordinate frame. and let $\mathbf{t}$ be the vector joining the origin of the camera frame to the origin of the world frame. The patch $\mathbf{x}$ is then given in the camera coordinate system by:

$$\hat{\mathbf{x}} = \mathbf{R}\mathbf{x} + \mathbf{t}. \tag{5}$$

Consider a pin-hole camera model (figure 2.a) with a coordinate system $(\mathbf{o}, \mathbf{i}, \mathbf{j}, \mathbf{k})$ attached to the camera such that the origin $\mathbf{o}$ is at the center of the image plane, $(\mathbf{i}, \mathbf{j})$ is a basis of that plane, and the focal point $\mathbf{f}$

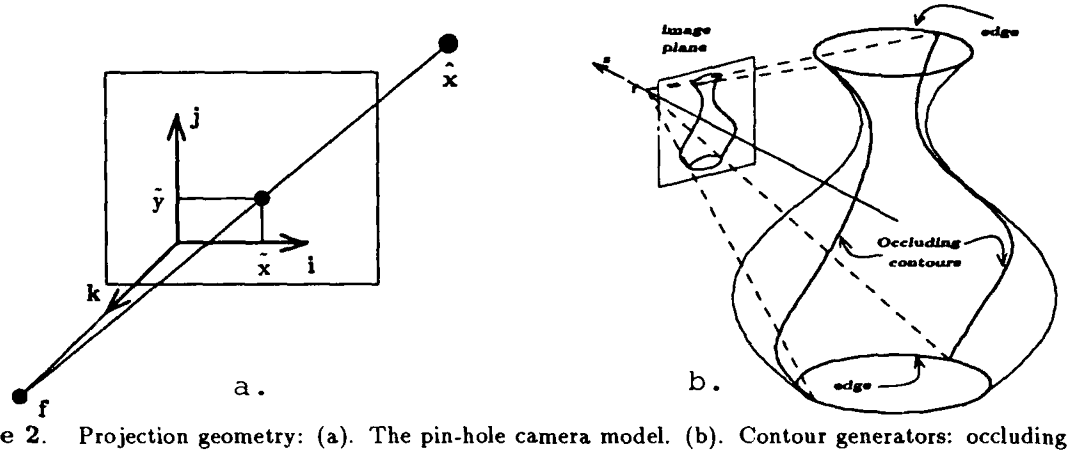


Figure 2. Projection geometry: (a). The pin-hole camera model. (b). Contour generators: occluding contours and edges.

is on the positive **k** axis, at a distance f from the origin. From now on, a point in the image plane is denoted by $\tilde{\mathbf{x}}$, with image coordinates $(\tilde{x}, \tilde{y})$. The image coordinates of a point on a patch in perspective projection are given by:

$$(\tilde{x}, \tilde{y}) = (\frac{f\hat{x}}{f - \hat{z}}, \frac{f\hat{y}}{f - \hat{z}}). \tag{6}$$

In the case of weak perspective projection (also called scaled orthography), all points on the same object are considered to have the same nominal depth, $\hat{z}_0$, so the above equation becomes:

$$(\tilde{x}, \tilde{y}) = (c\hat{x}, c\hat{y}), \tag{7}$$

where the constant c is given by $f/(f - \hat{z}_0)$.

Notice that in both cases, the projection is completely determined by six parameters: The rotation **R** can be parameterized by three angles (e.g., Euler angles), and its coefficients can be written as low degree polynomials in the cosines and sines of these angles. The translation **t** is given by its three coordinates. These six parameters completely determine the perspective projection. In the case of weak perspective, we have added an additional parameter c. Notice however that the third component of **t** is not used anymore, so the weak perspective projection can also be parameterized by six independent parameters. In the sequel, the viewing parameters will be denoted in both cases by $\mathbf{p} = (p_1, .., p_6)^t$. It is trivial that equations 6 and 7 can both be rewritten after substituting equations 5 and 1 and multiplying by the appropriate denominators as:

$$P_1(s, t, \tilde{x}, \mathbf{p}) = 0, \quad P_2(s, t, \tilde{y}, \mathbf{p}) = 0, \tag{8}$$

where P_1 and P_2 are polynomials in s, t, $\tilde{x}$, $\tilde{y}$, in the translation components of **p**, and in the cosines and sines of the rotation components of **p**.

3.2 CONTOUR GENERATORS

The following two types of contour generators are considered: edges, where the surface normal is discontinuous, and occluding contours, where the viewing direction is tangent to the surface (figure 2.b). The image contours are the projections of these two types of generators. In this section, explicit equations for the contour generators in the parameter space of a patch are obtained. Edges on a surface are view-independent contour generators. A given parametric patch may, a priori, have cusps where the surface normal is discontinuous [2]. However, in the context of CAD models, patches are usually built so that they do not have cusps [19], and the only edges are the intersection curves between different parametric patches. These curves are given by eq. 4, independently from the rigid transformation applied to the observed object.

Occluding contours are more complex, since they depend on the viewing direction. Trivially, the normal to a rational parametric patch is itself a parametric patch, given by:

$$\hat{\mathbf{n}}(s, t) = \frac{\partial \hat{\mathbf{x}}}{\partial s}(s, t) \times \frac{\partial \hat{\mathbf{x}}}{\partial t}(s, t), \tag{9}$$

where "×" denotes the cross-product. If **v** is the viewing direction, i.e., the direction of the line joining a point on the surface to the eye, occluding contours are given by:

$$\hat{\mathbf{n}} \cdot \mathbf{v} = 0. \tag{10}$$

Under weak perspective, the viewing direction is a constant $\mathbf{v} = (0, 0, 1)^t$. Under perspective, the viewing direction depends on the point observed, and is given by $\mathbf{v} = \tilde{\mathbf{x}} - \mathbf{f}$. By multiplying the edge or the occluding contour equation by the appropriate denominators, we obtain one more polynomial equation:

$$C(s, t, \tilde{x}, \tilde{y}, \mathbf{p}) = 0 \tag{11}$$

3.3 ELIMINATION

We can finally regroup the projection equations (eq. 8) and the contour equation (eq. 11) into a system of three equations:

$$\begin{cases} C(s, t, \tilde{x}, \tilde{y}, \mathbf{p}) = 0 \\ P_1(s, t, \tilde{x}, \mathbf{p}) \quad = 0 \\ P_2(s, t, \tilde{y}, \mathbf{p}) \quad = 0 \end{cases} \tag{12}$$

By eliminating s and t between these three equations, we finally obtain an expression of the form:

$$F(\tilde{x}, \tilde{y}, \mathbf{p}) = 0, \tag{13}$$

which is the implicit equation of the contours in the image. This equation is a polynomial in $\tilde{x}$ and $\tilde{y}$, in the translation parameters, and in the cosines and sines of the rotation parameters.

4. DETERMINING PARAMETERS FROM IMAGE DATA

Our goal is to find the set of parameters, $\mathbf{p}_0$ which best describes the location of the object from a monocular image. A measure of the error of fit for a particular set of parameter values and the image data must be minimized. The function F, obtained in eq. 13, defines an image curve and is zero for all contour points. Without any error in the edgels' location due to noise, edge detector bias, or precision, F would be zero. Computing the viewing parameters can therefore be reduced to the least squares problem of minimizing

$$\sum_i F^2(\tilde{x}_i, \tilde{y}_i, \mathbf{p}) \tag{14}$$

with respect to $\mathbf{p}$, where the $(\tilde{x}_i, \tilde{y}_i)$ are the observed contour points. However, the function F is *not* a measure of the distance between the theoretical contour and the data points, so direct minimization of the above sum can introduce some bias.

4.1 EXACT DISTANCE FUNCTION

Elimination can once again be used to obtain a closed-form expression of the distance between a point and a contour. Remember that the distance between a point $\tilde{\mathbf{x}}_i$ and a curve is defined as the minimum of the distance between this point and the curve points; this minimum is reached at a point $\tilde{\mathbf{x}}$ on the curve where the curve's normal and the line joining $\tilde{\mathbf{x}}$ and $\tilde{\mathbf{x}}_i$ are aligned. The distance is therefore given by the following system of equations:

$$\begin{cases} d^2 - (\tilde{x}_i - \tilde{x})^2 - (\tilde{y}_i - \tilde{y})^2 & = 0 \\ F(\tilde{x}, \tilde{y}, \mathbf{p}) & = 0 \\ (\tilde{x}_i - \tilde{x}) \dfrac{\partial F}{\partial \tilde{y}}(\tilde{x}, \tilde{y}, \mathbf{p}) - (\tilde{y}_i - \tilde{y}) \dfrac{\partial F}{\partial \tilde{x}}(\tilde{x}, \tilde{y}, \mathbf{p}) & = 0 \end{cases} \tag{15}$$

This time, the variables $\tilde{x}$ and $\tilde{y}$ are eliminated between these three equations leading to a new equation:

$$D(d, \tilde{x}_i, \tilde{y}_i, \mathbf{p}) = 0, \tag{16}$$

where D is a polynomial in the distance, d, and the parameters p_i. For a given transformation $\mathbf{p}$, the distance d is the minimum positive root of this polynomial, and can be found by some numerical root-finding algorithm [46]. Also, note that since d is given by an implicit equation, it is possible to compute its derivatives with respect to the viewing parameters, which is useful for numerical minimization.

4.2 APPROXIMATE DISTANCE FUNCTION

Computing the exact distance function is expensive, since it involves the elimination of two variables between three equations, two of which are of high degree, followed by numerical root finding. On the other

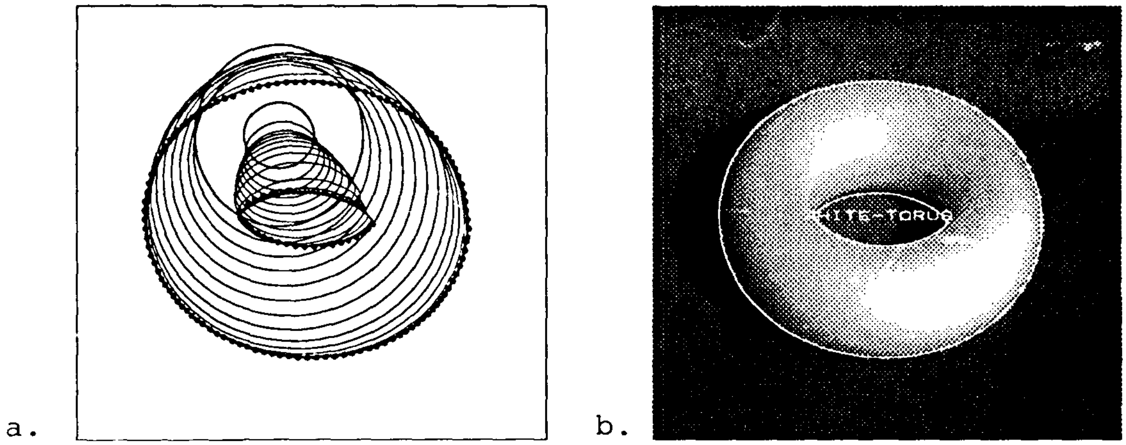


Figure 3. The result of minimizing the mean square error of the implicit contour equation: (a). The contours at each iteration of the minimization are shown overlaid on the edge points used in the minimization; the Canny edgels are drawn as little circles. (b). The result of recognition for the white torus.

hand, using only the function F to estimate the goodness of fit between data points and the computed silhouette can be biased. Because of this, we now propose another method to estimate the goodness of fit once the optimal parameters have been computed. As noticed before, the minimum distance between $\tilde{\mathbf{x}}_i$ and the contour is reached at a contour point $\tilde{\mathbf{x}}$ where the contour's normal is aligned with the line joining $\tilde{\mathbf{x}}_i$ and $\tilde{\mathbf{x}}$. If we suppose that the data points are a close fit to the curve, then the normal to the contour found in the image at $\tilde{\mathbf{x}}_i$ is approximately the normal to the corresponding theoretical contour. A reasonable edge detector [13] returns not only edgel location to subpixel accuracy but edgel direction as well. Let $\tilde{\mathbf{n}}_i$ be the normal to the edgel direction at $\tilde{\mathbf{x}}_i$, the minimum distance is reached at a point $\tilde{\mathbf{x}} = \tilde{\mathbf{x}}_i + \lambda\tilde{\mathbf{n}}_i$ such that $F(\tilde{x}, \tilde{y}, \mathbf{p}) = 0$. This can be rewritten as an equation in λ:

$$F(\tilde{x}_i + \lambda\tilde{n}_{x_i}, \tilde{y}_i + \lambda\tilde{n}_{y_i}, \mathbf{p}) = 0. \tag{17}$$

Since a good initial estimate of λ is available from fitting (i.e. $\lambda = 0$), a Newton-Raphson algorithm is used to find the first zero of this equation, and the distance is readily obtained. This method has been used in the experiments reported in the next section.

5. IMPLEMENTATION AND RESULTS

To demonstrate the feasibility of our approach, we chose to implement a simple recognition algorithm for a limited world made of tori of different sizes and flavors (five plastic rings from a baby's toy, a doughnut, and a bagel) observed under weak perspective projection. Why such a choice? Tori are obtained by sweeping a circle of radius r along another circle of radius R; they are at the same time simple enough (smooth surfaces of revolution that can be parameterized by a single biquadratic parametric patch) and complicated enough (non-convex surfaces with non-planar occluding contours) to qualify as initial test objects. In addition, tori can be characterized under weak perspective projection by a single shape parameter R/r, which allowed us to use a very simple matching algorithm, and yet to experiment with parameterized recognition.

5.1 IMPLEMENTATION

The approach described in this paper has been implemented as a three-step process. Implicit equations for the image contours of the models are first computed off-line; they are then fitted on-line to the measured contour points; finally models are recognized by comparing the errors of fit obtained. The derivation of the contour equation in the case of tori is detailed in [35]. Using the fact that a torus is a surface of revolution, it is possible to replace the elimination, as described in section 3.3, by the elimination of a single variable. This elimination step is done on a Symbolics lisp machine by using the Reduce implementation of the resultant of two polynomials. Note that several other algebraic manipulation systems are commercially available (e.g., Macsyma, Mathematica), and that they all offer some version of the resultant of two polynomials. An implementation of the Canny edge detector [13] is used to find contours. Because of shadows, surface markings, and irregularities in the surface and background, extraneous contours are found. They are eliminated by hand, and the remaining edgels are manually grouped into clusters corresponding to single tori. Better, automated segmentation is left for future work. A variety of numerical algorithms are needed for fitting the implicit contour equation to the data points and measuring the error of fit: we use the Levenberg-Marquardt algorithm to solve the nonlinear least squares minimization of eq. 14, and the safe Newton-Raphson algorithm to solve eq. 17 [46]. The Levenberg-Marquardt algorithm only finds a local minimum and requires a set of initial conditions. From a set of contours, global measures, such as center of gravity and moments along with extrema in the principal directions, are used to automatically determine a reasonable set of initial conditions. From a catalog of tori, it is possible to recognize which torus (or tori) is in an image. Two trivial approaches have been implemented: For a set of data points, try to fit all the possible models in the catalog; the recognized object is the model minimizing the image distance. Alternatively, the distinguishing feature of different tori is the ratio R/r. Parameterized recognition has been implemented, for a single "generic torus" model that contains this additional parameter. The parameterized torus is fit to the edges, and the resulting ratio is compared to those in the catalog.

5.2 RESULTS

The models used in our first set of experiments are five plastic rings, that will be referred to by their colors: red (R), white (W), blue (B), orange (O), and yellow (Y), even though they are really distinguished in our black and white images by their R/r ratios: 2.31, 2.44, 2.62, 2.67, and 2.83.

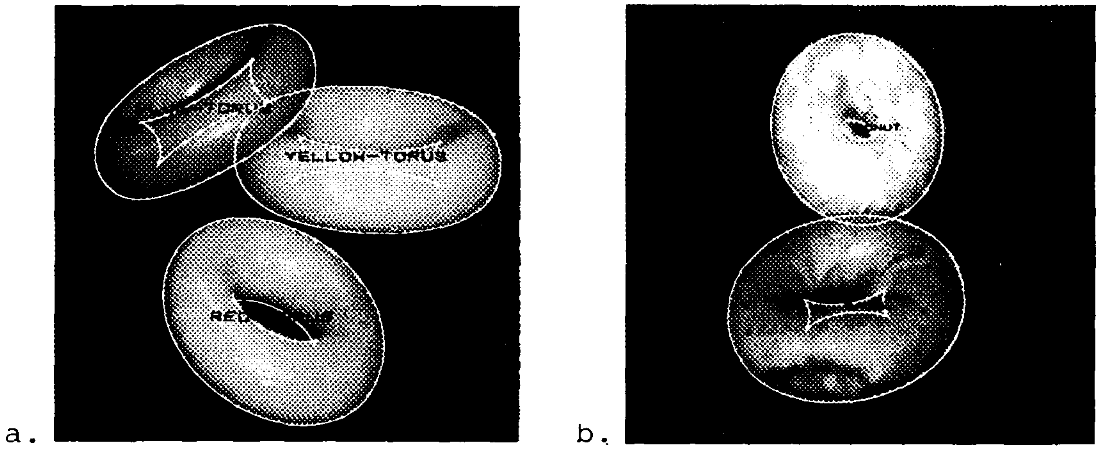


Figure 4. More recognition results: (a). Another image of three tori and the results of recognition. In this figure as in figures 1, 3, the recognized models are drawn as transparent objects without hidden-line removal (e.g., the lower portion of the yellow torus should actually be hidden in this view). (b). Generic recognition and positioning in the context of soft goodies: bagel vs. doughnut.

Figure 3.a shows the successive steps of the fitting of the W model for an image of that torus. Note that the five viewing parameters vary simultaneously. In our recognition tests, all five models are fitted, and the correct model (W) is recognized (figure 3). Figures 1 and 4.a show more complicated examples, with three different tori and occlusion. Again, the correct tori are recognized. The total computing times on a Symbolics lisp machine for these three examples are respectively 15, 33, and 32 seconds. Table 1.a summarizes the errors of fit obtained. Note that the average distance between the theoretical contours and the data points varies between 0.57 and 1.08 pixel for the recognized models.

Generic recognition, where the ratio R/r is also determined, has been performed for the three images. Visually, the results appear similar to those obtained by the first method, so their drawings are omitted. Quantitative results are summarized in table 1.b. The precision is slightly better using this method in the examples of figures 1 and 3, and slightly worse in the case of figure 4.a. Even though six parameters instead of five are estimated, this method is faster than the first one, because only one model is fitted to each set of data. The total computing times for our three examples are respectively 5, 14, and 15 seconds.

Generic recognition has been applied to distinguish a bagel from a doughnut (figure 4.a). These objects are rather poorly approximated by tori due to bumps, bruises, and other irregularities. However, the algorithm performs successfully and recognizes them, with an average error of about 1.5 pixel for both objects. The computing time is 10 seconds in this case.

Scene	Torus	R	W	B	O	Y
Fig. 1	O	4.29	2.96	1.12	0.72	0.95
	Y	4.70	3.87	2.16	1.69	0.57
	W	1.39	0.84	1.56	1.85	3.15
Fig. 3	W	2.67	1.08	1.90	2.49	4.32
Fig. 4	B	2.38	1.44	0.97	0.98	1.73
	Y	4.47	3.57	2.21	1.98	1.01
	R	1.02	1.50	3.04	3.59	4.75

a.

Scene	Torus	Th. R/r	Exp. R/r	Dist.
Fig. 1	O	2.67	2.71	0.29
	Y	2.83	2.83	0 28
	W	2.44	2.40	0.83
Fig. 3	W	2.44	2.48	0.57
Fig. 4	B	2.62	2.53	1.54
	Y	2.83	2.92	0.82
	R	2.31	2.37	1.18

b.

Table 1. (a). Summary of the recognition results. The leftmost column is the scene considered, the second column indicates which torus is observed in the scene, and the other columns give the average distances between the edge points and the model contours. The underlined distance in each row is the smallest distance, corresponding to the recognized model. (b). Summary of the results of generic recognition. The theoretical ratio, experimental ratio, and average image distance (in pixels) are given for the tori recognized in the different scenes.

6. CONCLUSIONS AND FUTURE RESEARCH

Elimination theory has been used to determine the implicit equation of the image contours of parametric patches. In turn, this equation has been used in an implemented algorithm for recognizing and positioning parameterized three-dimensional curved objects from their image contours. Although it has proven reliable and computationally efficient, the current implementation is limited to the world of tori, edge groups are hand-selected, and the matching algorithm is trivial. As shown in [35], the scope of the method proposed in this paper extends to most representations used in computer aided geometric design and computer vision. In practice, computing the implicit equation of the contours requires the elimination of two variables between three equations for general parametric patches, and the elimination of three variables between four equations for algebraic surfaces defined implicitly. This can, in principle, be done by recursive elimination of a single variable, but at the cost of adding extraneous terms. In addition, the degree of the resulting equations quickly becomes unwieldy [22]. Part of the problem is that elimination theory has mainly been considered a "black box" in this paper. To handle complex object classes as modelled in [34], it will be necessary to investigate other elimination techniques, such as resolvants [37] and multivariate resultants [12]. We have not really addressed the issue of control of the recognition process, and this is why we have had to rely on hand segmentation and very simple recognition strategies in our implementation. Future research will be dedicated to designing segmentation and recognition strategies suited to our positioning approach, in the same way as interpretation trees have been used in conjunction with rigidity constraints in the case of polyhedra [20,23,26,28,36,52].

Let us conclude by sketching a few other new applications of elimination theory to computer vision and graphics that we are currently investigating. Superquadrics are a popular representation for three-dimensional objects in computer graphics [2] and computer vision [1,24,42]. Segmentation of range images which contain superquadrics involves fitting the superquadrics parameters to the range data. As noted recently in [24], no good distance measure is known. Elimination can be used to solve that problem, as previously proposed by Buchberger [10], who was advocating the use of Gröbner bases for computing the distance between two superquadrics.

A fundamental problem in computer graphics is the generation of line-drawings of CAD models, with or without hidden-line removal. It is not possible to directly use the implicit equations of contours for that purpose. However, the contours can be drawn by finding the extremal and singular points, and using numerical marching techniques between them. This has been done in CAD systems for building trimmed surface patches [19], but not, as far as we know, for actually drawing solid models. The singular points are given by the following set of equations:

$$F(\tilde{x},\tilde{y}) = 0, \frac{\partial F}{\partial \tilde{x}}(\tilde{x},\tilde{y}) = 0, \frac{\partial F}{\partial \tilde{y}}(\tilde{x},\tilde{y}) = 0, \tag{18}$$

and they can be found by eliminating $\tilde{x}$ and $\tilde{y}$ between these equations. Moreover, the visibility of the contours only changes at cusps and T-junctions, which are exactly the singular points, so hidden lines can be removed for no additional cost.

Aspect graphs, introduced a decade ago by Koenderink and Van Doorn [33], are back in fashion (see, for example, [21,25,29]). Visual events have been described for smooth objects [11,47], but very few attempts have been made at actually computing the aspect graphs of these objects [17]. Once again, the aspects can be characterized by the singular points of the contours. Approximate aspect graphs can then be found by tessellating the Gaussian sphere and grouping identical aspects. Future research will be concerned with the construction of exact aspect graphs by similar methods.

Acknowledgements: We wish to thank John Canny and Martial Hebert for useful discussions and comments. Support for this work was provided in part by the Air Force Office of Scientific Research under contract F33615-85-C-5106 and by the Advanced Research Projects Agency of the Department of Defense under Knowledge Based Vision ADS subcontract S1093 S-1. David Kriegman was supported in part by a fellowship from the Fannie and John Hertz Foundation.

References

[1] R. Bajcsy and F. Solina. Three-dimensional object representation revisited. In *Proc. of the International Conference on Computer Vision*, pages 231–240, London, U.K., June 1987.

[2] A. Barr. Superquadrics and angle preserving transformations. *IEEE Computer Graphics and Applications*, 1:11–23, January 1981.

[3] H. Barrow and J. Tenenbaum. Interpreting line drawings of three-dimensional surfaces. *Artificial Intelligence*, 17(1-3):75–116, 1981.

[4] P. Besl. Geometric modelling and computer vision. *Proceedings of the IEEE*, 76(8):936–958, 1988.

[5] P. Besl and R. Jain. Three-dimensional object recognition. *ACM Computing Surveys*, 17(1):75–145, 1985.

[6] T. Binford. Generic surface interpretation: observability model. In *Proc. of the 4^{th} International Symposium on Robotics Research*, Santa Cruz, August 1987.

[7] R. Bolles, P. Horaud, and M. Hannah. 3DPO: a three-dimensional part orientation system. In J. Brady and R. Paul, editors, *Proc. of the 1^{st} Inteinational Symposium on Robotics Research*, pages 413–424, MIT Press, Cambridge, Ma., 1984.

[8] J. Brady and A. Yuille. An extremum principle for shape from contour. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, PAMI-6:288–301, 1984.

[9] R. Brooks. Symbolic reasoning among 3D models and 2D images. *Artificial Intelligence*, 17:285–348, 1981.

[10] B. Buchberger. Applications of Gröbner bases in non-linear computational geometry. In R. Janssen, editor, *Trends in computer algebra: international symposium*, pages 52–80, Springer-Verlag, 1987.

[11] J. Callahan and R. Weiss. A model for describing surface shape. In *Proc. of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 240–245, 1985.

[12] J. Canny. *The complexity of robot motion planning*. MIT Press, 1988.

[13] J. Canny. *Finding lines and edges in images*. MIT AI Lab. TR-720, Massachusetts Institute of Technology, 1983.

[14] M. Clowes. On seing things. *Artificial Intelligence*, 2(1):79–116, 1971.

[15] D. Cyrluk, D. Kapur, and J. Mundy. Algebraic reasoning in view consistency and parameterized model matching problems. In *Proc. of the DARPA Image Understanding Workshop*, pages 731–739, Boston, Ma., April 1988.

[16] A. Dixon. The eliminant of three quantics in two independent variables. *Proc. London Mathematical Society, Series 2*, 7, 1908.

[17] D. Eggert and K. Bowyer. *Computing the orthographic projection aspect graph of solids of revolution*. Comp. Sc. Tech. Rep., University of South Florida, December 1984.

[18] T. Fan, G. Médioni, and R. Nevatia. Matching 3D objects using surface descriptions. In *Proc. of the IEEE Conference on Robotics and Automation*, pages 1400–1406, Philadelphia, Pa., April 1988.

[19] R. Farouki. Trimmed-surface algorithms for the evaluation and interrogation of solid boundary representations. *The IBM Journal of Research and Development*, 31(3):314–333, 1987.

[20] O. Faugeras and M. Hebert. The representation, recognition, and locating of 3-d objects. *The International Journal of Robotics Research*, 5(3):27–52, Fall 1986.

[21] Z. Gigus, J. Canny, and R. Seidel. Efficiently computing and representing aspect graphs of polyhedral objects. In *Proc. of the International Conference on Computer Vision*, pages 30–39, Tampa, Fl., December 1988.

[22] R. Goldman and T. Sederberg. Some applications of resultants to problems in computational geometry. *The Visual Computer*, 1:101–107, 1985.

[23] W. Grimson and T. Lozano-Perez. Localizing overlapping parts by searching the interpretation tree. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, PAMI-9(4):469–482, 1987.

[24] A. Gross and T. Boult. Error of fit measures for recovering parametric solids. In *Proc. of the International Conference on Computer Vision*, pages 690–694, Tampa, Fl., December 1988.

[25] M. Hebert and T. Kanade. The 3D profile method for object recognition. In *Proc. of the IEEE Conference on Computer Vision and Pattern Recognition*, San Francisco, CA., June 1985.

[26] R. Horaud. New methods for matching 3-D objects with single perspective views. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, PAMI-9(3):401–412, 1987.

[27] D. Huffman. Impossible objects as nonsense sentences. *Machine Intelligence*, 6:295–323, 1971.

[28] D. Huttenlocher and S. Ullman. Object recognition using alignment. In *Proc. of the International Conference on Computer Vision*, pages 102–111, London, U.K., June 1987.

[29] K. Ikeuchi. Precompiling a geometrical model into an interpretation tree for object recognition in bin-picking tasks. In *Proc. of the DARPA Image Understanding Workshop*, pages 321–339, Los Angeles, Ca., February 1987.

[30] J. Kajiya. Ray tracing parametric patches. In *Proc. of SIGGRAPH 82*, pages 245–254, July 1982.

[31] T. Kanade. Recovery of the 3-D shape of an object from a single view. *Artificial Intelligence*, 17(1-3):409–460, 1981.

[32] J. Koenderink. What does the occluding contour tell us about solid shape? *Perception*, 13, 1984.

[33] J. Koenderink and A. V. Doorn. The internal representation of solid shape with respect to vision. *Biological Cybernetics*, 32:211–216, 1979.

[34] D. Kriegman and T. Binford. Generic models for robot navigation. In *Proc. of the IEEE Conference on Robotics and Automation*, Philadelphia, PA, April 1988.

[35] D. J. Kriegman and J. Ponce. *On recognizing and positioning curved 3D objects from image contours.* Robotics Laboratory Internal Report, Stanford University, 1989. To appear.

[36] D. Lowe. The viewpoint consistency constraint. *The International Journal of Computer Vision*, 1(1):57–72, 1987.

[37] F. Macaulay. *The algebraic theory of modular systems.* Cambridge University Press, 1916.

[38] J. Malik. Interpreting line drawings of curved objects. *The International Journal of Computer Vision*, 1(1):73–103, 1987.

[39] D. Marr. Analysis of occluding contour. *Royal Society of London*, B-197:441–475, 1977.

[40] V. Nalwa. Line-drawing interpretation: bilateral symmetry. In *Proc. of the DARPA Image Understanding Workshop*, pages 956–967, Los Angeles, Ca., February 1987.

[41] R. Nevatia and T. Binford. Description and recognition of complex curved objects. *Artificial Intelligence*, 8:77–98, 1977.

[42] A. Pentland. *Recognition by parts.* Technical Note 406, SRI International, Menlo Park, CA., 1986.

[43] J. Ponce. On characterizing ribbons and finding skewed symmetries. In *Proc. of the IEEE Conference on Robotics and Automation*, Scottsdale, Ar., May 1989.

[44] J. Ponce and D. Chelberg. Finding the limbs and cusps of generalized cylinders. *The International Journal of Computer Vision*, 1(3), October 1987.

[45] J. Ponce, D. Chelberg, and W. Mann. Invariant properties of straight homogeneous generalized cylinders and their contours. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 1989. Accepted for publication.

[46] W. Press, B. Flannery, S. Teukolsky, and W. Vetterling. *Numerical recipes in C.* Cambridge University Press, 1986.

[47] J. Rieger. On the classification of views of piecewise smooth objects. *Image and Vision Computing*, 1:91–97, 1987.

[48] G. Salmon. *Modern higher algebra.* Hodges, Smith, and Co., Dublin, 1866.

[49] T. Sederberg, D. Anderson, and R. Goldman. Implicit representation of parametric curves and surfaces. *Computer Vision, Graphics, and Image Processing*, 28:72–84, 1984.

[50] S. Shafer. *Shadows and silhouettes in computer vision.* Kluwer Academic Publishers, 1985.

[51] K. Stevens. The visual interpretation of visual contours. *Artificial Intelligence*, 17(1-3):47–73, August 1981.

[52] K. Sugihara. An algebraic aproach to the shape-from-image-problem. *Artificial Intelligence*, 23:59–95, 1984.

[53] F. Ulupinar and R. Nevatia. Using symmetries for analysis of shape from contour. In *Proc. of the International Conference on Computer Vision*, pages 414-426, Tampa, Fl., December 1988.

Computer Vision Research

At the IBM T. J. Watson Research Center[†]

Ruud M. Bolle, Andrea Califano, Rick Kjeldsen, and Russell W. Taylor
Exploratory Computer Vision Group
IBM Thomas J. Watson Research Center
PO Box 704, Yorktown Heights, NY 10598

Abstract

We will describe the research activities in the Exploratory Computer Vision Group at the IBM T. J. Watson Research Center. The current research is summarized and the future directions are outlined.

The main focus of the ongoing work is the development of an experimental vision system for recognition of 3D objects. A homogeneous architecture is proposed that supports recognition of simple partial features to complex feature assemblies and 3D objects. At all levels of recognition, the same techniques are used, namely, *parameter transforms* and *recognition networks*. Parameter transforms generate hypotheses in recognition networks, motivated by connectionist systems, which fuse evidence from various sources and insure global consistency. Within this framework, we have implemented a system which extracts surface patches and surface intersection curves from a depth map. These reconstructed features index into an object database to find consistent interpretations.

Introduction

Work at the Exploratory Computer Vision Group has centered around a system intended to recognize complex 3D objects in cluttered environments. To support this work, we have proposed a homogeneous *framework* for recognition.

Figure 1 represents an overview of our approach. The recognition task is structured as a hierarchy of layered and concurrent *parameter transforms* [1] for feature extraction. Features that are structurally independent, for instance, planes and linear 3D edges, form concurrent paths of recognition. Features that depend upon other lower-level features, for instance, boxes and planar patches, are placed in hierarchical layers within a path. Parameter transforms generate votes for hypotheses in parameter spaces. Evidence from various sources is fused in a manner motivated by work in *connectionist* vision systems [6,9]. The result is a highly parallel, highly modular architecture using a uniform control structure.

The most important aspect of our approach is the homogeneity that allows different feature types (such as, surfaces and curves) and potentially different input sources (range data, intensity data, tactile data, ...) to be easily integrated. A global interpretation of the scene is arrived at through the fusion of diverse sources of evidence. This homogeneity is obtained with the introduction of a *generalized concept of feature* which allows features at any level to be treated uniformly.

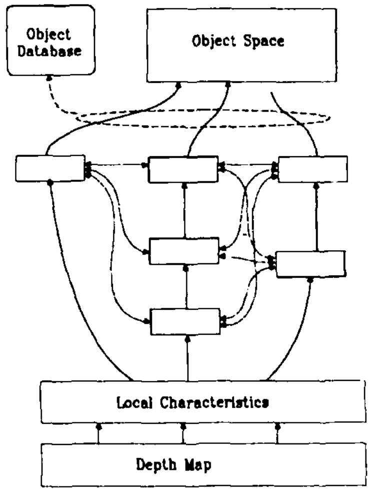


Fig. 1 System Architecture

Generalized features

The Generalized Feature Concept imposes a uniformity on the features in a system, allowing them to interact. This, in combination with the control strategy to be discussed, makes possible the integration of evidence from diverse features. The concept states that each feature type is defined by a parameterization and a set of relationships to other features.

[†] This paper discusses only the work done at the Computer Science Department of the IBM T. J. Watson Research Center. In addition to this work, there is computer vision research done at the Manufacturing Research Center of the lab. See, for example, [7,8,12].

In order to define a feature, we establish the particular parameterization, along with procedures to reach this parameterization from lower level features ,*parameter transforms*), and to establish evidential relationships with other features at any level of any path (*compatibility relations*). A given feature can be introduced into the system simply by specifying these three types of knowledge.

To aid discussion of the feature hierarchy in the system, we subdivide the term feature into *local, partial, primitive* and *assembly* features, in order of increasing level of abstraction from the original data. For an example consider lines in three-space. The line is a *primitive* feature contained in many higher level feature *assemblies* such as parallelpipeds. But it is also formed from *partial* features such as orientation and position, which, in turn, are extracted from *local* features such as the output of an edge detector.

Organization of recognition

The generalized feature concept allows a homogeneous, feature independent control structure. This takes the form of a recognition network [6,9] where nodes represent feature hypotheses and links represent the evidential relationships between features. Parameter hypotheses for a feature are collected in a *parameter space* associated with that feature type. Each parameter space is a subnet of the recognition network. Parameter transforms map from some input parameter space into some other parameter space [1] and accumulate evidence for feature hypotheses in a manner similar to the Hough transform. Iterative refinement, similar to that used in connectionist systems, allows the network to determine which hypotheses are actually present.

The links in the network are (1) bottom-up connections between local features and the nodes, and (2) links between nodes themselves. The latter links can be inhibitory, in case the hypotheses are conflicting, or excitatory, in case the hypotheses are supporting one another. Hypotheses and links are generated dynamically from the results of the parameter transforms and compatibility relations.

Each node computes an activation level representing the confidence in the existence of the corresponding feature or object in the input. At each iterative step *i*, the activation level of a node, denoted by $AL_{node}(i)$, is computed as

$$AL_{node}(0) = 0$$
$$AL_{node}(i) = AL_{node}(i-1) + BU_{node} + LE_{node}(i-1) - LI_{node}(i-1) - D \ ,$$

where BU_{node} represents bottom-up reinforcement, that is, a measure of confidence that the corresponding hypothesis exists based only on data measurements (see [10]). LE_{node} *and* LI_{node}, represent lateral excitation or inhibition from other hypotheses. D is a decay term that helps suppress spurious hypotheses. If a unit in a space survives, i' votes for hypotheses in higher level spaces via the associated parameter transform.

This approach provides several advantages for a recognition system.

Because of the uniform nature of the architecture, the parameter transforms for a feature can be defined so that evidence can be gathered from different input sources while the control paradigm remains unchanged. This creates a natural environment for evidence fusion.

The ability of the system to accept data from a variety of sources, and to utilize a rich set of features, creates a beneficial redundancy. For instance, a planar surface and its bounding edges supply redundant information that allows for more robust recognition of bounded planar patches than either feature alone. In combination with the highly parallel architecture, this implies that failure or removal of any one fe. ture extraction path need not imply failure of the recognition process as a whole.

Inhibition links perform several useful functions. Inhibition within a small parametric neighborhood sharpens the response of the transforms. Only the strongest unit in any neighboring cluster will survive. This reduces the problem of votes for a hypothesis being split between several buckets in parameter space. Inhibition links between hypotheses which are supported by common image pixels provide an implicit segmentation of the image Only those hypotheses which do not share support from portions of the image will survive iteration. These will represent a spatial segmentation of the image. *LI* links can also be used to ensure that hypotheses which are inconsistent, for example, for geometric reasons, will compete.

The evidence integration process is inherently noise resistant. Most hypotheses receive support from many sources of evidence. Therefore missing or inaccurate evidence will only cause a small change in the total amount of support for a hypothesis.

A vision system

Using these techniques we have developed a system capable of recognizing objects in depth maps of complex scenes. This section will describe the features we use and the parameter transforms used to extract them. For a more complete description see [3].

System features

Our surface feature set consists of planes, spheres and quadrics of revolution, specifically cylinders, and cones. To increase coverage we also include curves in three-space, namely lines and conic-sections. These correspond to intersections and boundaries of surface patches.

It is important to note that our system has no dependency on any specific choice of features. Thanks to the generalized feature concept, as we expand the system's scope, features can be incrementally added or the feature set substantially changed with little effort.

Multiple window parameter extraction

To extract the parameters of complex geometric entities, one would like to devise an $M \times M$ operator that computes some parametric description of the curves and surfaces. To avoid interference from nearby local features, the size M of the operator should be small–but this will make estimates of higher-order properties of the curve and surfaces inaccurate.

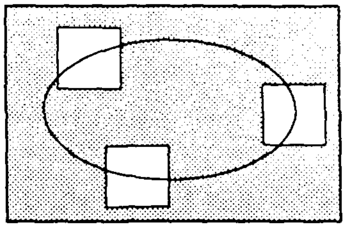


Fig. 2: Multiple windows for finding ellipses.

To solve these problems, we use the correlated information embedded in different windows that contain portions of a feature. For both curve and surface extraction, we use a set of nearby range points or windows to examine a more global neighborhood and extract the parameters of our primitive features. Specifically, for each range point **q**, we use all possible combinations of n windows (within some *radius of coherence*) to generate hypotheses about the presence of features in the scene. Though many spurious hypotheses are generated, only those actually present collect sufficient evidence to survive. Hence, we extend the pure local parameter extraction to a somewhat more global process of parameter estimation. This is illustrated in Fig. 2 for the case of ellipse finding using multiple windows. (The multiple window approach is described in detail in [4,5].)

Feature reconstruction

Local surface features are extracted from smooth surface approximations to the depth map. That is, least-squares second order polynomial approximations [2] are made within $M \times N$ areas about range point **q**. From these approximations, the principal curvatures, κ_{max} and κ_{min}, and the associated principal directions in three-space, $\mathbf{X}_{max}$ and $\mathbf{X}_{min}$, for each range point q are computed.

We define 3D points that lie on depth discontinuities as local curve features. Well-known edge detectors are used to generate discontinuity maps; this gives range points q on, or near zero and first order depth discontinuities.

These local features are used as input to the first level parameter transforms in the recognition hierarchy. For example, consider two range points $\mathbf{q}_a$ and $\mathbf{q}_b$ that lie on a quadric of revolution (not an oblate ellipsoid). Then, the plane P_a containing $\mathbf{q}_a$ and spanned by $\mathbf{X}^a_{min}$ and the normal $\mathbf{N}^a$ contains the axis of revolution. Similarly, for plane P_b spanned by the normal at point $\mathbf{q}_b$ and the direction of minimum curvature. Hence, the intersection of these two planes gives us a hypothesis for the axis of revolution.

When a first level hypothesis survives in its winner-take-all subnetwork, it is used as input to a parameter transform to determine higher level feature hypotheses. These transforms generally reexamine the image points, $\{\mathbf{q}_n\}$, which support the hypothesis. For example, the radius R associated with each point in $\{\mathbf{q}_n\}$ of a sphere center hypothesis $\mathbf{p}$ is given by $R = \|\mathbf{p} - \mathbf{q}\|$.

Object recognition

The homogeneous approach of the system is maintained for object recognition as well. The parameter transform from the feature spaces to object space uses the reconstructed features to index into a database of object models to determine which it suggests.

We have no need to determine which features belong to a single object *before* recognition. Just as the refinement step provides an implicit segmentation of the image during feature extraction, it also provides a partitioning of the features during object recognition. We specify that hypotheses which share support from common features compete, thus assembly hypotheses which survive evidence integration will not share features, and a partition on the feature set is created.

Object models

Object models in the database are represented by a feature graph (see, for example, Fig. 3). Nodes in the graph represent the primitive features of the object; surfaces and their intersection curves. Arcs represent *coordinate-free* geometric relationships between features, for example, the relative size of the radii of a sphere and cylinder.

Features of a model are sorted into layers, which represent the resolution at which we expect them to be reconstructed. The first layers contain those features likely to be found at a coarse resolution and successively later layers contain progressively finer features. This "multiple resolution" representation prunes the search for matching features as will be described below.

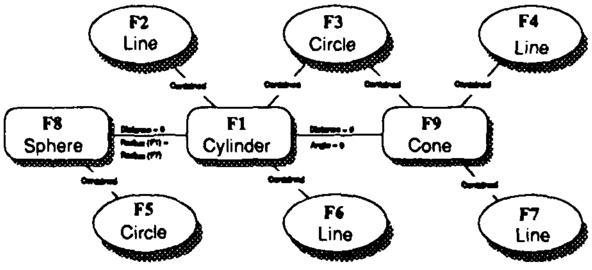


Fig. 3: Feature graph of an object.

Each hypothesis in object space represents an instantiation of an object model from the database. It is identified by the set of bindings $\{B(F_i,f_j), \ldots\}$ between features F_i of the object model and features f_j found in the image.

Indexing

Indexing consists of two steps, checking for a match with unbound model features of existing object hypotheses and checking features of models in the database. If a match is found we extend or create a hypothesis to include the new evidence.

In either case, we only need to check for matches between image features and features in *active* layers of an object model. For uninstantiated models in the database, only the first (coarsest resolution) layer is active. Instances of models in object hypotheses can have one or more active layers. Layers are activated whenever sufficient features of the previous layer have been bound to image features. To avoid an explosion in hypotheses with very little evidence, only models with sufficient matches in the first layer are instantiated. Matching an image feature to a feature in a model requires checking two pieces of information, intrinsic feature characteristics (e.g., feature type) and position relative to other features.

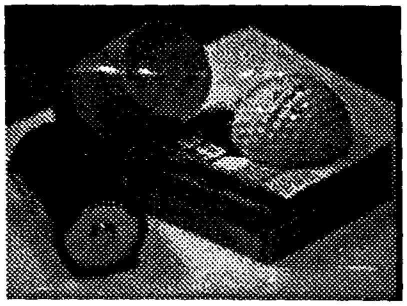


Fig. 4: Complex real-world scene.

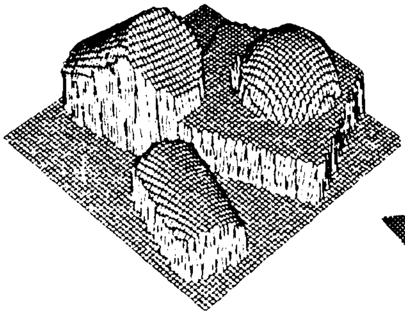


Fig. 5: Depth map.

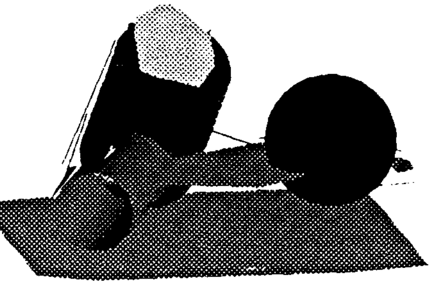


Fig. 6: Reconstructed Features

Experiments

Experiments have been run on some twenty images of varying complexity. Early images were artificially generated and relatively simple (see, e.g., [2]). More recently, we have been using images generated with a laser range finder [11]. Images have varied from 32 × 32 to 256 × 256.

We present an experiment using a 64 × 64 depth map generated from the scene in Fig. 4. It contains four simple objects; a pencil sharpener, a battery, a tape box, and a half golf ball. One end of the pencil sharpener and the golf ball are resting on the box. Figure 5 shows the depth map.

Low level processing includes finding the zero and first-order discontinuities in the original data, and computing quadric smooth surface approximations about each point. With this size image, these take about two minutes on a Symbolics 3650. Feature extraction takes on the order of 12 iterations and 15 minutes real time. Most surface and discontinuity features present in the image were successfully reconstructed, cf. Fig. 6.

Evidence integration causes a dramatic implosion in the number of hypotheses, as can be seen in Fig. 7, which shows the number of hypotheses in each parameter space over time. As an example, 17 axis orientation hypotheses (partial features of axes of quadrics of revolution) were generated in the first-level quadric of revolution reconstruction process. Of these two survived and generated 89 hypotheses for axes of revolution. Only the two correct axes survived to reconstruct the solids of revolution, i.e., hypotheses for cylinders and cones.

For this experiment, there were 11 object models in the database, each containing from two to 20 features (averaging 12) and from one to 110 relationships (averaging 34). Model features were divided into two layers. The models of the objects in the image were a simple sphere (two features, spherical surface and a circular limb), a cylinder segment (7 features and 8 relationships), and a box (18 features and 63 relationships). Other models in the database included a bottle, an L-bracket and two different computer mice.

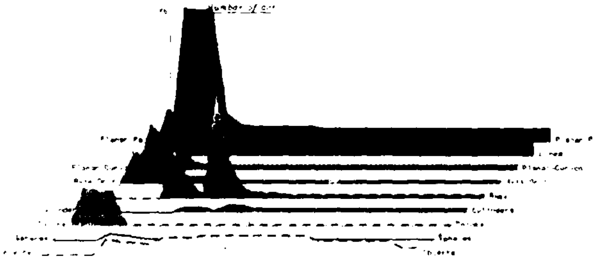


Fig. 7: Number of active hypotheses over time.

The four objects were all successfully identified. The large cylinder segment was identified on the basis of five features, the cylindrical surface, the bounding end plane, the limbs and the circle formed by the intersection of the bounding plane and the cylinder. The smaller cylinder, sphere and box hypotheses were similarly bound to all their reconstructed features. The total indexing time for all features was on the order of four minutes.

In an attempt to demonstrate noise resistance, we incrementally added noise to the image of the previous experiment and, without changing any control parameters, observed the behavior of the system. With white Gaussian noise of standard deviation $\sigma = 0.3$ pixels (or 1% of the dynamic range of the image) we began to fail to recognize some features. Performance then degraded gracefully till, with noise of standard deviation $\sigma = 0.6$ pixels (2% of the dynamic range) (Fig. 8) roughly half the features in the image were no longer reconstructed and only two of the objects were recognized. In separate tests we were able to improve performance in noisy images substantially by adjusting the control parameters.

Research directions

A number of areas have been identified as current and future research topics for extending the system. The general thrust is to scale system performance with input scenes of increasing complexity and size, as well as object model databases containing more, and more complex objects.

Object modeling

Building object models and organizing these models in an object model database is a difficult and important problem; this becomes more of an issue when the catalogue of objects is large. Additionally, efficient schemes for indexing into the database become important as image complexity increases.

Often an object can be modeled as the composition of subobjects which suggests hierarchical modeling. However, for large object domains, an hierarchical organization of the database is not enough. The presence of a particular feature in the scene will still generate an object hypothesis for every possible object or subobject in the database that has that feature as a composing part. Therefore, we need a scheme that limits the explosion of interpretations. A commonly suggested solution to this is an multiple resolution approach to object modeling and matching. For example, one could introduce an ordering of the features and describe an object by adding more and more detail. We have generalized this concept and propose to use any ordering of the features. To be more precise, an object is described in terms of layers of sets of features and the layers are ordered. Matching and indexing is achieved starting at the first layer, progressing to the next layer, and so on. Examples of layering criteria are:

- Ordering the features according to how "well they describe" an object. If the modeling is done in terms of surface and curve patches, this would be multiresolution modeling, describing the objects in a coarse to fine manner.
- An ordering of the features in terms of those features that "best distinguish" the object from other objects in the database.

Of course, many other criteria for layering the object features are possible. We propose to use several different layerings of the object features. It is conceivable that in the early stages of the recognition process, prominent objects in a scene are ıecognized by their coarse features, while in later stages other salient features are important. Note that each layer of features in an object model describes a subassembly which not necessarily corresponds to a physical object. Hence, objects are described in terms of subobjects and subassemblies. Each subobject can be part of many objects, while on the other hand, each subassembly can compose many objects, that is, many objects can share the same layers.

One has to specify how the features within the subobjects and subassemblies are related and how these constructs are related to one another. Here we consider two choices, the use of an object-centered coordinate system and the use of coordinate-free relationships between the primitive features and constructs. Most relationships can be described using either of these methods. However, some relationships are more naturally described with one than with the other. The advantage of coordinate-free relationships is that they can be checked very efficiently but defining unique and exhaustive relationships between all pairs of features of a construct requires large amounts of storage (the number of relationships grows exponentionally with the number of features). On the other hand, the use of object-centered coordinate systems is computationally more expensive but allows for a complete specification of all geometric relations between (geometric)

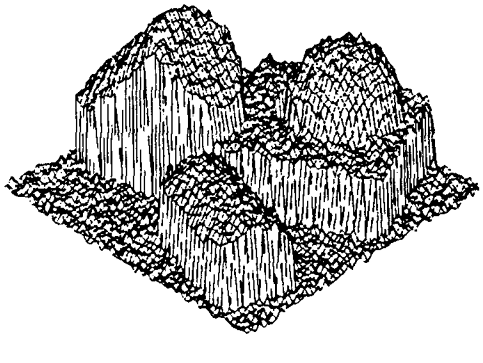


Fig. 8: Degraded depth map.

features and constructs relatively easily. Therefore, we propose a compromise:

- All features in all constructs and all constructs of an object are related to each other in terms of an object-centered coordinate system. Hence, once the transformation between the object-centered coordinate system and a reference frame defined v.ith respect image features is known, all other (geometric) features can be matched to the object in terms of this transformation.
- Some features, typically, but not necessarily features in the first layers (subassemblies), of an object are related in a graph structure as described above. This allows for very fast indexing into the database since only simple geometric relations have to be checked during this process, as opposed to computing rigid transformations.

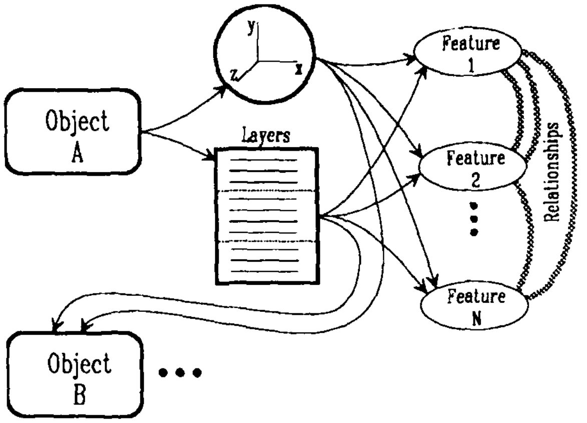


Fig. 9: Expanded object modeling scheme.

Figure 9 gives an example of an object model. Object A is described in terms of a subobject (object B) and a number of features. The features, including the subobject, are ordered in layers; some of the features are related by coordinate-free relationships while all the features are described within an object-centered reference frame.

Of course, with such complicated models, automatic model acquisition becomes a challenging task.

Multiresolution

We are investigating techniques to reduce overall data processing requirements. Our approach will involve a progression from coarse to fine image resolution. The cues for going to a finer resolution (and hence more data) will be derived from the lower level data or data descriptions, and from the higher level model description spaces. By organizing the models themselves into a multiresolution representation, we will be able to implement *model-driven* feedback to the sensing stages, to perform a foveation-type search for salient features. Similarly, lower level information will produce a more *data-driven* feedback to the sensing stage.

Our model database contains information on which object components are fine features, and which are coarse features. An initial search of a low resolution image will instantiate a set of object model hypotheses based on coarse features. The models themselves can then be examined for distinctive fine features, and more data may be examined in appropriate regions to try and detect this information. Thus, for example, a bolt and a battery might be essentially cylinders at coarse resolution, but a model-driven search at the ends of the cylinder would yield additional fine planar features in the case of the bolt (its head).

Resolution changes driven by lower level information are a more *ad hoc* process, since we are generally trying to infer where we are missing information. Reasonable paradigms are to increase resolution to achieve more accurate localization of discontinuities, and to improve surface parameter estimation. We are investigating two techniques:

- The limitations of the fitted surface can be used as the driving source for resampling. For example, in case of local least-squares surface fitting, if the error is excessive compared to the expected image noise level, then a finer resolution may be appropriate.
- Guided by the above technique, we start processing and image at a certain coarse resolution. Regions of the image that contain coarse surface patches, will quickly converge to a solution. Regions that contain surface discontinuities or fine surface detail will not converge or converge much slower. These latter regions can be resampled and processed at a finer resolution.

Let us give an example of the second approach. Parameter reconstruction of, for example, the cylinders in the depth map of Fig. 8 fails because the signal to noise ratio in the cylindrical areas is too high. There are no curve or surface hypotheses associated with these (known!) areas and processing these areas at twice the resolution will help the system to identify the surfaces.

Fusion

We wish to add more features to the system to increase the range of objects that can be recognized. As with any differentiation and identification task, the more orthogonal the descriptors are, the better the result. Various types of features would be reasonable and interesting to add to the system. Of particular interest is the class of features that can be obtained from intensity imagery. These may include texture and color. Also current features, such as discontinuities, could be augmented by intensity data analysis. Because of the nature of our architecture, such diverse features can easily be added to the system.

An additional data-driven path would let intensity image analysis drive the more expensive range imaging process, where local analysis would call for finer resolution to resolve ambiguous surface regions.

Discussion

We have described a framework for visual recognition. Within this framework we have implemented a vision system that recognizes objects bounded by planar and curved surfaces using a depth map as input. The experiments show that the proposed techniques are potentially very powerful.

We intend to further develop this system to determine whether the approach scales with the complexity of the problem.

References

[1] D.H. Ballard, "Parameter nets: A theory of low level vision," in *Proc. 7th Int. Joint Conf. On Artificial Intell.*, Aug. 1981, pp. 1068-1078.

[2] R.M. Bolle, R. Kjeldsen, and D. Sabbah, "Primitive shape extraction from range data," in *Proc. IEEE Workshop on Comp. Vision*, Nov.-Dec. 1987, pp. 324-326; also IBM Tech. Rep. RC 12392, July 1987.

[3] R.M. Bolle, A. Califano, R. Kjeldsen, and R.W. Taylor, "Visual Recognition Using Concurrent and Layered Parameter Networks," in *Proc. IEEE Conf. on Comp. Vision and Pattern Recog.*, June 1989; also IBM Tech. Rep. RC 14249, Dec. 1988.

[4] A. Califano, "Feature recognition using correlated information contained in multiple neighborhoods," in *Proc. 7th Nat. Conf. on Artificial Intell.*, July 1988, pp. 831-836; also IBM Tech. Rep. RC 14116, July 1988.

[5] A. Califano, R.M. Bolle, and R.W. Taylor, "Generalized neighborhoods: A new approach to complex feature extraction," in *Proc. IEEE Conf. on Comp. Vision and Pattern Recog.*, June 1989; also IBM Tech. Rep. RC 14305, Jan. 1989.

[6] J.A. Feldman and D.H. Ballard, "Connectionist models and their properties," *Cognitive Science*, Vol. 6, 1981, pp. 205-254.

[7] R.K. Lentz and R.Y. Tsai, "Calibrating a Cartisian robot with eye-on-hand configuration independent of eye-to-hand relationship," in *Proc. IEEE Conf. on Comp. Vision and Pattern Recog.*, June 1988, pp. 67-75; also IBM Tech. Rep. RC 13211, Oct. 1987 (to appear in *IEEE Trans. on Pattern Analysis and Machine Intell.*, Sept. 1989).

[8] R.K. Lentz and R.Y. Tsai, "Techniques for calibration of the scale factor and image center for high accuracy 3D machine vision metrology," *IEEE Trans. on Pattern Analysis and Machine Intell.*, Vol. 10, No. 5, Sept. 1988, pp. 713-719.

[9] D. Sabbah, "Computing with connections in visual recognition of origami objects," *Cognitive Science*, Vol. 9, No. 1, Jan.-March 1985, pp. 25-50.

[10] D. Sabbah and R.M. Bolle, "Extraction of surface parameters from depth maps viewing planes and quadrics of revolution," in *Proc. SPIE Conf. Intell. Robots and Comp. Vision*, Oct. 1986, pp. 222-232.

[11] Technical Arts Corp., *110X 3D Scanner: User's manual & application programming guide.* Redmond, WA: 1986.

[12] R.Y. Tsai, "A versatile camera calibration technique for high-accuracy 3D machine vision metrology using off-the-shelf TV cameras and lenses," *IEEE Journal of Robotics and Automation*, Vol. 3, No. 4, pp. 323-344.

An Experimental Target Recognition System for Laser Radar Imagery*

Dan E. Dudgeon Jacques G. Verly Richard L. Delanoy
M.I.T. Lincoln Laboratory
Machine Intelligence Technology Group
P. O. Box 73
Lexington, Massachusetts 02173-0073

Abstract

The automatic recognition of targets is an important and difficult problem. The targets to be recognized may appear in a variety of conformations and they may appear against a variety of backgrounds. We have developed a model-based experimental target recognition system to process laser radar range and intensity imagery and to recognize targets automatically. This system is a laboratory research tool; it allows us to try alternative algorithms and vary parameters while evaluating their effect on the overall recognition performance of the system. In the course of our research we have exercised the system on several hundred images to obtain the performance results discussed in this paper.

1 The ATR Problem

The goal of an automatic target recognition (ATR) system is to detect and recognize various objects of interest in a particular area using imagery or other measurements from one or more sensors. In addition the ATR system may have (and will probably need) access to other information such as map data, navigation data, suspected target locations, suspected target types, and meteorological data. Ideally, the output of the ATR system would be a prioritized list of targets, their locations, orientations, configurations, and important attributes.

The ATR problem is unquestionably difficult. An object of interest may appear different when viewed from different viewing angles, in different states of articulation, when surrounded by different backgrounds, at different times of day, under different weather conditions, when imaged by different sensors, when partially obscured, and when camouflaged. Backgrounds can vary dramatically, making it difficult to build a system that gives an acceptably low false alarm rate under all conditions. On its surface, the ATR problem is "simply" one of detecting and classifying signals in noise. However, the signals and noise have widely varying characteristics, and the signal, when it is present, is not simply added to the noise.

The Department of Defense is very interested in ATR systems for a number of applications. Some of these applications, such as the tactical battlefield, involve areas containing a relatively large number of targets. and others, such as holding mobile missile launchers at risk, are akin to looking for a needle in a haystack. The use of a reliable, robust ATR system will reduce the workload of the pilot and crew in a manned aircraft and will reduce the bandwidth of data links needed by remotely piloted or

*This works was supported by the DARPA Tactical Technology Office. The views expressed are those of the authors and do not reflect the official policy or position of the U.S. Government.

autonomous reconnaissance aircraft. There is also interest in developing "smart" weapons, which can be launched from stand-off positions and which will search and engage certain targets in specified areas. We are interested in the tactical air-to-ground situation, where an aircraft must detect, identify, and prioritize the targets it encounters. (In spite of this interest, the bulk of our initial work has been done using ground-based imagery because of its availability.) The problem of how the disparate pieces of sensed and *a priori* information are represented in the computer is central to the solution of the ATR problem. It is the ability of the recognition system to apply the right information to the problem at the right time that will make its solution possible. Numerical data are easily represented and manipulated by computers. Other forms of information, which are less precise, less clearly related, less prone to mathematical analysis, must also be represented and manipulated to solve the ATR problem. The symbolic representation of information and transformations between various symbolic representations are fundamental issues in the development of ATR systems.

2 Model-Based Recognition

In addressing the ATR problem, we have concentrated on a model-based approach to object recognition. This approach generally consists of four major elements, as indicated in Figure 1. In the *image event characterization subsystem*, imagery from sensors is analyzed by one or more algorithms to detect, extract and represent information about the objects in the scene. This operation is often referred to as "feature extraction," but that name is somewhat misleading and limiting. This subsystem is more like a set of image analysis tools that can be applied selectively to parts of an image or the entire image as dictated by the ATR control subsystem. The event characterization subsystem not only performs the image processing computations necessary for analysis, but it represents the results of the analysis in a form that is suitable for the matching operation to follow. Ideally, the output of the event characterization

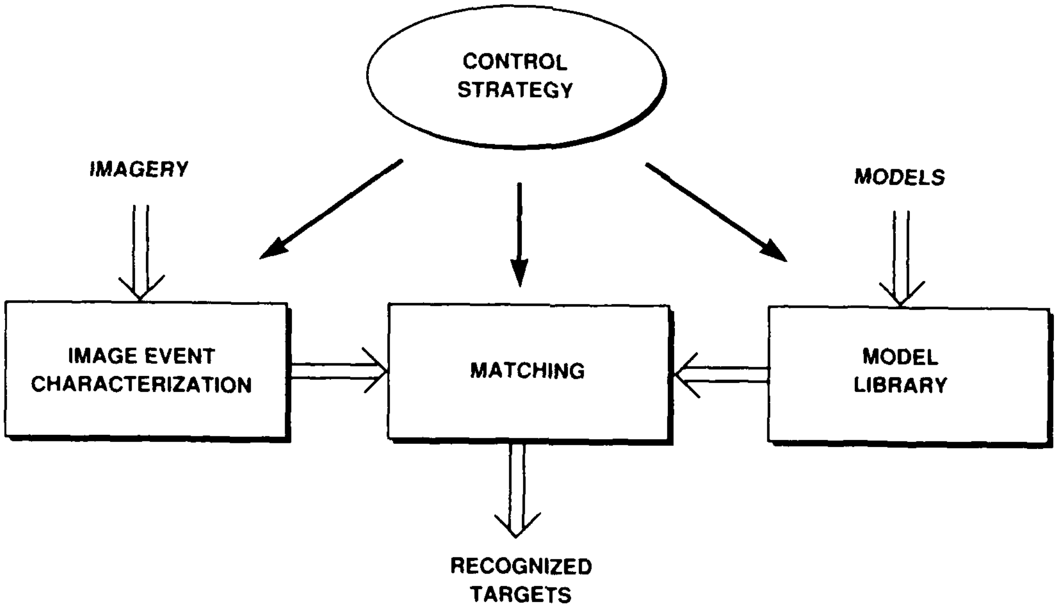


Figure 1: Block diagram of a general model-based recognition system.

subsystem would be a list of interesting areas of the image (detections) and a characterization of each of these areas using both symbolic and numeric data representations.

The *model library* contains explicit models of objects and backgrounds organized to facilitate the recognition process. Here lies a great challenge, to encode the diversity of *a priori* information known about objects, the environment in which they exist, and their relationships to one another in a way that can be used to match the information obtained by the image event characterization subsystem. The contents of the model library also depend on the capabilities of the sensors, because it makes no sense to model what cannot be measured. Having this knowledge in an easily accessible form facilitates development and makes it easier to understand what the system is doing when it recognizes an object.

Currently models are built "by hand," that is, an analyst studies the imagery and the blueprints of the objects to be recognized and encodes the relevant information in a data structure that forms part of the model library. Some simple software tools are currently used in this task, but there is much more that could be done to develop an efficient computer-aided library building system. It is also possible, at least in some cases, to develop a program that can generate a model library *automatically* from many examples of objects in imagery. A simple version of automatic library generation has been demonstrated in a companion recognition project [Aull 88].

Between the image event characterization subsystem and the model library lies the *matching subsystem*. It attempts to match the characterizations of events detected in the image to the various object models. In essence it attempts to generate hypotheses for the identification of objects found in the image, discard those hypotheses found to be inconsistent with the imagery, compute a figure of merit for the strength of the match between the extracted features and the model for each consistent hypothesis, and make a decision based on the figures of merit.

Indexing is an important task for the matching subsystem. In this context, indexing refers to generating a set of working hypotheses based on partial characterizations of an object detected in the imagery. To recognize an object with a unique and persistent feature, often called a "signature," it is sufficient to detect that feature in the imagery. The feature is a strong index into the library of possible object hypotheses. More generally, however, any particular feature will be exhibited by several types of object. The feature is a weaker index in this case, but nevertheless it serves to reduce the number of hypotheses that are consistent with the observed imagery.

Development of a practical matching subsystem is a challenge. Conceptually it is straightforward to conceive of a matching subsystem that examines all models (from a finite library of models) for each object detected in the image and then attempts to evaluate the strength of the match for each possibility. This corresponds to mapping out all points in a space of possible solutions and exhaustively testing each one for each unknown object. Computationally, however, this strategy is generally impractical. It is more desirable to structure the matching operation so that many hypotheses can be eliminated with a relatively small amount of computation. We shall examine one approach for doing this later in this article.

The final major subsystem in a generic model-based recognition system is the *control subsystem*. Its primary job is to apply computational resources to the recognition problem to reduce the overall amount of computation needed to reach a reasonable decision. If computational resources, memory, and execution time were not limited, the control subsystem could simply conduct an exhaustive search of the solution space for every object detected in the image. Unfortunately, this highly structured, inflexible approach is impractical for all but the simplest problems.

To reduce the amount of computation expended, the control subsystem must embody strategies about when various analysis operations should be applied to the imagery and under which conditions various matching procedures should be invoked. The "intelligence" of the recognition system, which is still very much mechanical, depends on the variety, sophistication, and effectiveness of this procedural knowledge.

Two major control strategies, data-driven and model-driven, are usually present in a model-based recognition system. The data-driven control strategy starts from the bottom up by analyzing the input imagery, extracting features from it, grouping features into higher-level features, and attempting to associate the high-level features together to form objects. In the model-driven control strategy, an object model is used to guide the processing by dictating what specific features should be searched for in the image to determine the presence and location of that kind of object. Model-driven control is often called "top-down."

The conventional wisdom is that a robust ATR system will need to use both control strategies. Data-driven control is appropriate for detecting image events and extracting features from the imagery for forming hypotheses from extracted features. Model-driven control is appropriate for finding evidence to support or refute a hypothesis. The challenge becomes one of developing a mixed strategy that is able to distinguish between the situations appropriate for data-driven control and model-driven control and to embed this procedural knowledge into the control subsystem.

3 Experimental Target Recognition System

Over the past several years we have developed a model-based ATR system for recognizing tanks, howitzers, and armored personnel carriers (APCs) in laser radar imagery. Our goal was to create an end-to-end system, one that accepted laser radar range and intensity images as its input and produced a recognition decision as its output. We did not require that each piece of the system be optimal or state-of-the-art, but we did want to have enough pieces in place to comprise a complete system. With a complete system, we are able to conduct experiments with different subsystems, algorithms, and parameters to evaluate their impact on the end-to-end recognition performance.

We have been working primarily with imagery taken from a ground-based infrared laser radar system developed by the Opto-Radar Group at Lincoln Laboratory[Gschwendtner 83]. Laser energy is transmitted out into the environment using a pulse waveform. Reflected energy returns to the sensor and is detected to give a waveform such as the one shown in Figure 2. The peak intensity is a measure of the reflectivity of the object encountered by the laser radiation, and the peak location in time is a measure

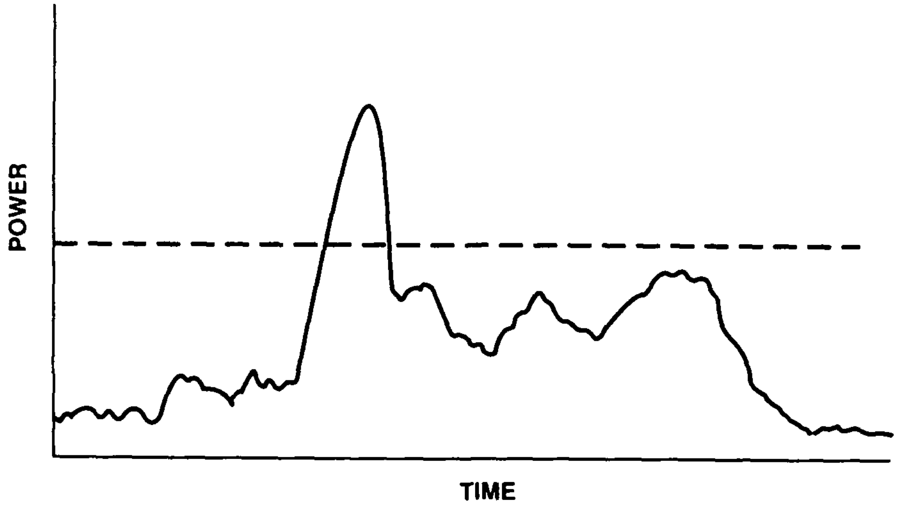


Figure 2: Typical laser radar returned waveform.

of the range to the reflecting object. These two measurements are made for each picture element (pixel) in an image by scanning the laser beam across the scene.

Other measurements may also be made with a laser radar sensor. By using a continuous waveform, Doppler shift may be measured to indicate the degree of radial motion in each pixel. Passive infrared detectors have also been installed using the same optics to provide a thermal image that is pixel-registered with the range and intensity images. A detailed discussion of laser radar technology, however, is outside the scope of this article.

Typical intensity and range images are shown in Figures 3 and 4 respectively.

In Figure 3 the top row of three intensity images shows three views of a tank, the middle row three views of an APC, and the bottom row three views of a howitzer. A bright pixel in an intensity image indicates a strong reflector of laser radiation, and a dark pixel indicates that very little energy was returned to the sensor from that area of the scene. Figure 4 shows the corresponding, pixel-registered range images. In range images, pixel brightness corresponds to distance from the sensor. Bright pixels are far away; dark pixels are close.

Note that these range images contain numerous "noise" pixels which appear blacked out. These "dropouts" correspond to areas of the scene that reflect very little radiation. This situation causes the returned signal to remain below the detection threshold. The sensor cannot reliably make a range estimate, and it records that fact.

Another type of noise is also present. Range pixels called "outliers" result when noise causes the wrong peak to be picked in the returned signal. Of course, in this case the sensor is not aware that the wrong peak was picked and records an erroneous range value for the pixel. Some outliers are visible in Figure 4 as white dots on the black area at the bottom of the images. Dropouts and outliers are collectively called *missing values.*

Our primary data base consists of 200 such intensity-range image pairs. Each image is 60 pixels high by 128 pixels wide. The images are zoomed vertically by a factor of two so that objects appear with the correct aspect ratio. Each image pair exhibits a tank, a howitzer, or an APC at one of four ranges: 700, 1000, 1400, and 1800 meters. The orientation of the vehicle, and in the case of tanks and howitzers the body-turret angle, may vary from one image pair to the next. Various subsets of the images have different background characteristics, including distant uncluttered backgrounds, near uncluttered backgrounds, and near cluttered backgrounds.

The sensor used to acquire our primary data base has a range accuracy of approximately 6 meters. Because of this, most pixels on an object fall into a single range bin or straddle the boundary between two adjacent range bins. In addition to this imagery, airborne laser radar imagery acquired with the Opto-Radar Group's improved range-accuracy sensor is also being examined.

We have also augmented the image database by developing a synthetic image generator. Using it, we can construct scenes and situations for which we do not have real imagery. It also allows us to know exactly the contents of the scenes we process and to vary sensor parameters, such as range accuracy, angular resolution, and percentage of outliers.

Our real laser radar imagery is characterized by coarse range resolution and a significant amount of dropout and outlier noise. The objects that appear in the imagery are often not known precisely; they may be partially occluded, have articulated parts rotated to various orientations, or have missing or extraneous parts.

The problem of recognizing 3-D objects from range imagery has received a lot of attention in the last few years [Kanade 87]. However, many published approaches assume high angular-resolution and high range-resolution range images in well controlled situations. Furthermore, the objects that are to be recognized are generally known precisely and rigid (as opposed to articulated). In our application, however, we need to perform recognition of imprecisely known and articulated targets using a 2-D silhouettes. Thus, most of the above approches are not well adapted to our problem.

The problem of recognizing 3-D objects from 2-D silhouettes extracted from the range imagery has

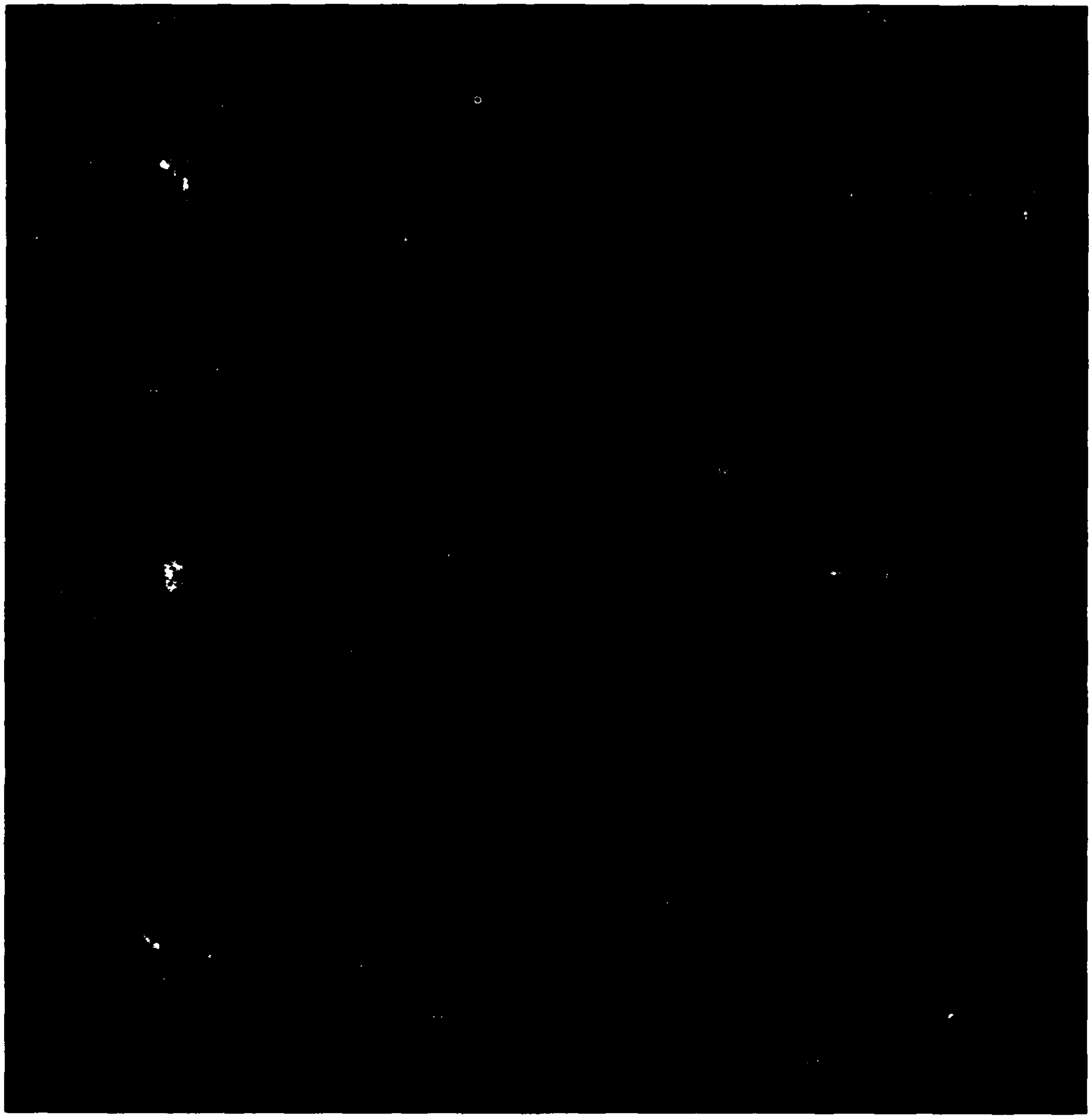


Figure 3: Laser radar intensity images.

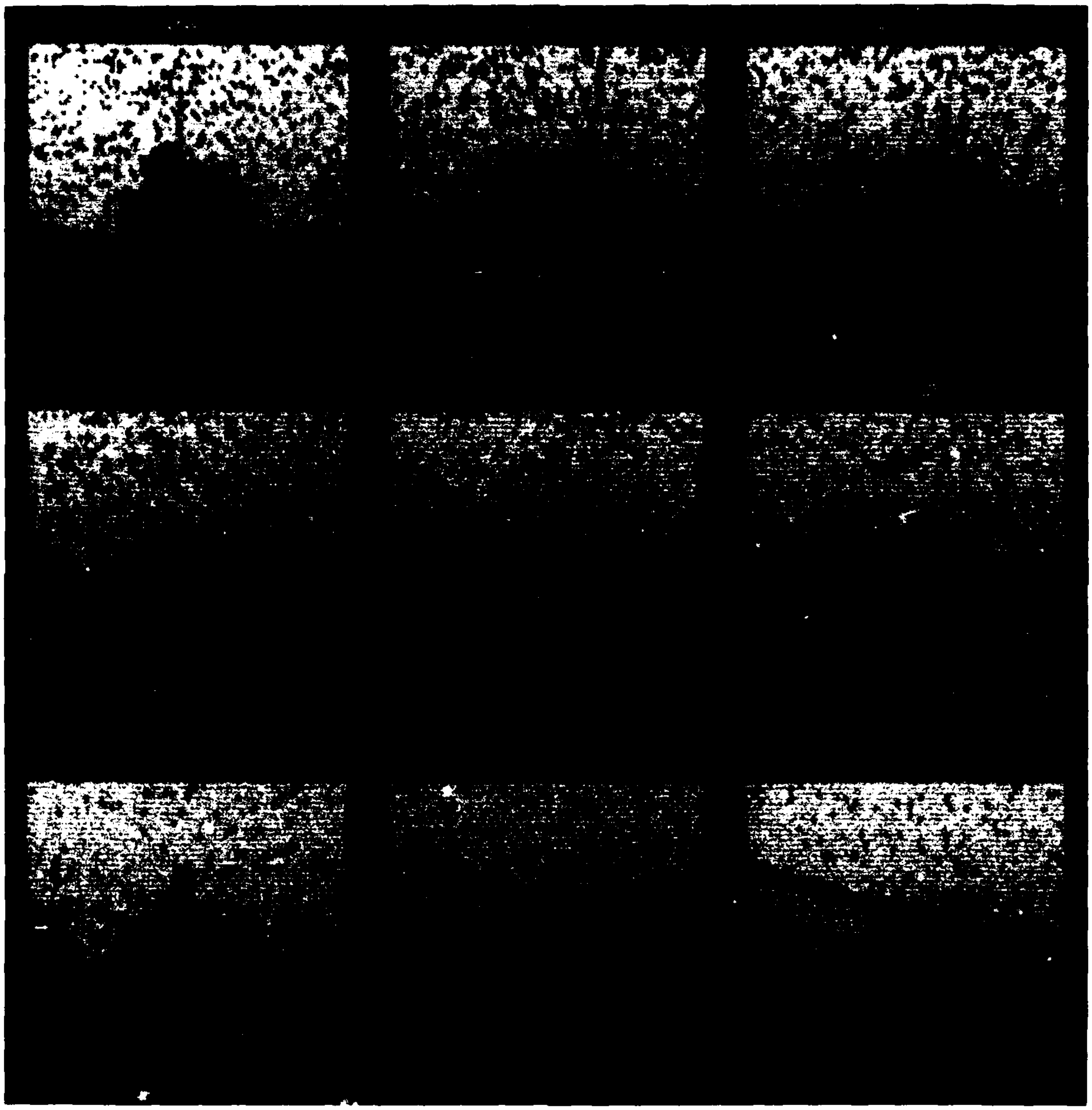


Figure 4: Laser radar range images.

not been widely studied. Recently Van Hove [Van Hove 87,Van Hove 88] developed a constraint-based approach to recognizing 3-D objects from edge segments of their 2-D silhouettes. However, detailed models and relatively clean silhouettes are required in his current recognition system. Work remains to test and extend his system to actual laser radar imagery.

The distinguishing features of XTRS are its ability to recognize possibly articulated, imprecisely known objects viewed from a variety of vantage points.

A block diagram of the experimental target recognition system (XTRS), shown in Figure 5, reflects the general block diagram discussed earlier, with a few variations. There are actually two distinct recognition systems within XTRS. Both extract range silhouettes from laser radar imagery and make recognition decisions automatically, but they differ in their processing approaches. The contour-based system (XTRS-C) attempts to extract range discontinuities corresponding to object boundaries while the region-based system (XTRS-R) attempts to extract regions having constant range. XTRS-C and XTRS-R differ only in their event characterization subsystems and model libraries; they use the same matching subsystem.

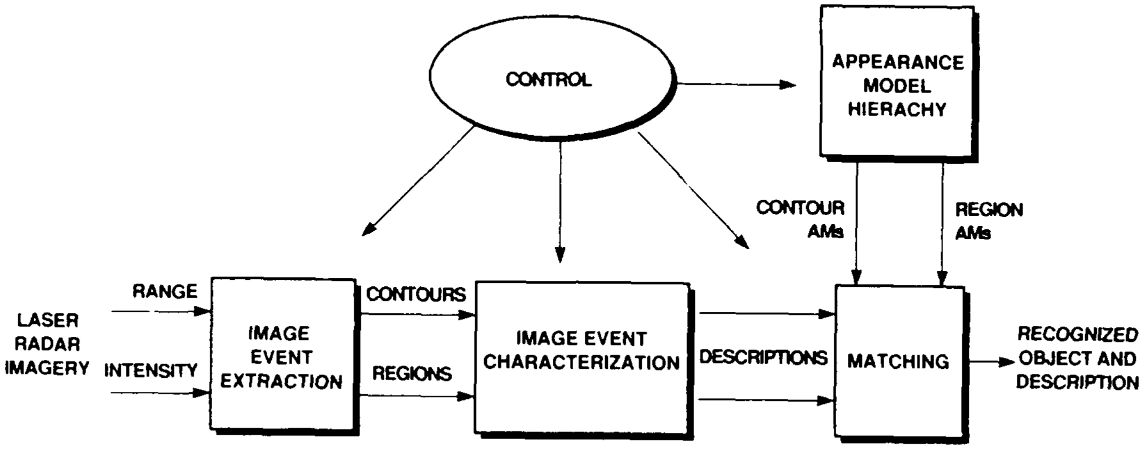


Figure 5: XTRS block diagram

The model libraries of XTRS use *appearance models* (AMs) to store information about each type of object that the system is to be able to identify. An AM, rather than being a full 3-D model, is a description of the expected appearance of an object in a 2-D image produced by a given sensor. A single AM describes the appearance of an object over a range of aspects and articulation conditions. An AM is also designed to allow missing and extraneous parts as well as some degree of occlusion. Each AM is a data structure, similar to a semantic network, that describes an object in terms of the sizes and shapes of its parts and the relations among them. The concept of an AM was first introduced by Selfridge [Selfridge 82], but our definition, implementation, and use of AMs differs from his.

In both the contour-based and region-based systems, the recognition step consists of matching the symbolic silhouette characterization against AMs of known objects and tl n deciding among the alternative objects (including the null hypothesis) using scores computed in the matching process.

In the following sections we shall discuss in greater detail the mechanisms by which the various subsystems are implemented in XTRS.

4 Image Event Characterization

The first step in processing the laser radar imagery is to detect, extract and characterize various *image events*. An image event is some particular characteristic that occurs in the imagery and may indicate the presence of an object. In XTRS-C, we attempt to find long contours representing range discontinuities. In XTRS-R, we try to extract regions with constant range values. In both cases the image events (*i.e.*, contours and regions) are characterized and stored as *symbolic image-event descriptions*. (The description is called symbolic though the characterization contains both symbolic and numeric attributes.)

An image event is also decomposed into primitive features. For XTRS-C these primitives are corners and arcs; for XTRS-R these primitives are subregions. A *symbolic primitive description* of each primitive is created to be used in matching against the symbolic descriptions embodied in the AMs. The image-event description and the primitive descriptions comprise the characterization needed for matching.

The primitive descriptions corresponding to an image event are stored in a data structure called an *attributed relational graph*, or *ARG* [Eshera 86]. An ARG is similar to a semantic network; it is a graph whose nodes contain symbolic and numeric attributes describing the primitive features and whose links between nodes indicate relationships between the features.

In the following subsections we shall describe in turn the event characterization procedures for XTRS-C and XTRS-R.

4.1 Contour Extraction and Characterization

The XTRS-C relies exclusively on the raw range images produced by the laser radar; the intensity images are not used. The first step is to remove as many "noise" pixels as possible. Each dropout is reported automatically by the sensor; outliers must be detected by computation. They are subsequently replaced with "locally reasonable" range values. (The detailed procedure for detecting and replacing missing values is described in [Verly 86].) The finai cleaned range image is obtained by removing remaining isolated "black" pixels using function (gray-scale) mathematical morphology [Serra 82]. (The procedure is based on the noise removal techniques of [Esselman 87a,Esselman 87b].)

A difference-of-Gausssians (DOG) edge detection algorithm [Marr 82] is used to extract contour information from the cleaned range image. The zero-crossings (ZCs) of the resulting image correspond in theory to the edges in the cleaned range image. The zero-crossing contours of the DOG image are formed by linking small ZC elements obtained by interpolating between positively valued and negatively valued pixels. Each of these contours has a number of associated attributes such as length, average strength, complete lists of range values on the near/far side (low/high range values) of the contour, as well as an indication of the position of the near side with respect to the contour.

Since the number of resulting contours is large and since the object of interest may yield several disjoint contours, a heuristic method was devised to merge contours into potential object silhouettes. We retain only the N contours with the largest average ZC strength. (We generally use $N = 20$.) Most of the pieces of the desired silhouette are generally present in this reduced set of contours. The complete silhouette contour can often be obtained by starting with the longest of these strongest contours, splitting some of the remaining open contours at places where the associated list of low range values shows significant discontinuities, splicing the resulting open contours to the longest contour (or each other), and finally reconnecting selected closed contours. (The details of these contour merging operations are complex and beyond the scope of this article.) The median value of the elements in the corresponding list of low range values is used as the object range value.

At this point in the processing we have selected a contour to be analyzed; the extraction part of the processing is complete. Now we must characterize the shape of the contour by decomposing it into its primitives. A condensed representation of the silhouette is obtained by computing a polygonal approximation (PA) to its contour. To do this, we have implemented Ramer's algorithm [Ramer 72].

The aim in computing this PA is to reduce the number of points used to represent the contour while retaining the essential shape of the silhouette. Each line segment in the PA corresponds to a subcontour of the silhouette and this correspondence information is kept as one of the attributes of the PA for subsequent use.

The PA is smoothed by removing vertices corresponding to small shears and small angular deviations. This step is called PA editing. The resulting edited PA can in some cases be identical to the original PA.

The edited PA is decomposed into concave and convex parts, and each of these edited-PA subcontours is further decomposed into corners and arcs (contour primitives). The decomposition procedure is similar, but not identical, to that of Pavlidis [Pavlidis 79], and its complete description is beyond the scope of this article. Since each corner or arc consists of two or more edited-PA segments, it is straighforward to find what part of the original silhouette corresponds to any given primitive. This ability to "backproject" contour primitives onto the extracted silhouette contour can be exploited in the recognition process, for example, to measure the median width of an elongated appendage using the extracted contour rather than its edited PA. The bounding rectangle of the edited PA is the frame of reference used during recognition.

Finally symbolic descriptions are constructed for the extracted silhouette contour and each of the primitives. In the case of the primitives, the descriptions include information such as length and orientation. The descriptions of the silhouette and the primitives are later matched against object contour AMs to make recognition decisions.

4.2 Region Extraction and Characterization

The XTRS-R processes simultaneous, spatially-registered raw intensity and range images. Except for omitting the final morphological processing, the raw range image is cleaned by the procedure described in the previous subsection. (See [Delanoy 89] for a detailed discussion of region extraction and characterization.)

Once cleaned, a range image is analyzed to identify those range bins containing high concentrations of features that are "interesting." For our purposes, "interest" is treated as a quantifiable attribute of an image; the higher the interest value for an area of the image, the greater the probability that it contains a target being sought. We construct *interest images* based on measured quantities to detect object regions.

Currently XTRS-R uses four interest images: the raw intensity image (the objects of interest should have one or more areas that produce strong laser radar returns), an image whose values indicate the presence of height-limited vertical surfaces (the sides of military vehicles are usually vertical surfaces no more than 3 meters in height), a "rod" image indicating the presence of rod-like objects (such as gun barrels and antennas), and a "body" image favoring objects with lengths between 2 and 8 meters. Other features, for example passive infrared signatures or motion, might also be useful in detecting object regions.

Interest values are summed for each range bin, forming a histogram. Local maxima in the histogram are used to form image segments. The segment with the largest combined interest sum is selected for further analysis. If the range to the selected segment is so large that the target will fill only a small portion of the image, a window (subimage) containing the highest concentration of interest values in the selected range interval is created.

At this point a trinary image is constructed. In over-simplified terms, a value of 1 is assigned to each pixel location that has a range value in the range interval defining the segment. A value of 0 to pixels with a greater range, and a value of 2 is assigned to pixels with a smaller range. Then pixels of value 2 are reset to 0 or 1 using a nearest neighbor algorithm. The resulting binary image generally consists of a number of disconnected 1-valued regions containing the object of interest, the ground, and some noise above the object. Noise is removed by complementing each 4-connected region of 0-valued elements which is entirely contained within a region of 1-valued elements. Set (binary) mathematical morphology [Serra 82] is then used to remove ground clutter and to reconnect vertical and horizontal appendages (the procedure used is similar to those described in [Esselman 87a,Esselman 87b]). All 8-connected regions of 1-valued elements are labeled to distinguish any distinct regions, and the distinct region with the largest area satisfying the constraints of the object class "military vehicle" is chosen as the object silhouette candidate. The median range value of the elements in the silhouette region is used as the object range value.

At this point in the processing, we have selected a region to be analyzed (the image event); we have completed the detection task. Now we must characterize the shape of the region by decomposing it into its major subregions. The region decomposition algorithm works on the outline of the silhouette region to obtain a contour. As bef.re, a polygonal approximation to the contour is computed using the algorithm of Ramer [Rамer 72]. The PA is edited by removing small intrusions and extrusions corresponding to surface rregularities; vertices corresponding to small angular deviations are also removed in a way similar to that described in the previous subsection.

The edited PA is then decomposed into subpolygons by selecting a set of line segments called "baselines." Baseline candidates that connect any concave vertices, or convex vertices that form nearly horizontal line segments, are collected. From these, a final set of baselines are selected using two sequentially applied rule bases. The first isolates rod-shaped extensions from the silhouette. Once these have been removed, a second set of rules identifies long, nearly horizontal baselines such as those at the junction of turret and body.

Finally, symbolic descriptions are constructed for the extracted silhouette region and each of the primitives. Attributes such as length, width, and orientation are calculated for each subregion. Symbolic and numeric descriptions of each subregion are arranged in an attributed relational graph (ARG) that represents the spatial relationships between subregions. Figure 6 shows a simple example of a region ARG.

This information is used by the matching subsystem to recognize the target.

5 Appearance Models

Appearance models (AMs) describe the expected appearances of objects in images. (See [Verly 89a] for a full description of the concept and construction of appearance models.) AMs are not full 3-D models; instead, they are an attempt to describe all possible appearances of an object over a limited range of viewing angles. In the present system, each AM describes an object in terms of its parts and the relations among them. The parts chosen for description in the model are those believed to be identifiable given the type of imagery and the means of extracting objects and their parts from images. These models are represented using data structures that are similar to semantic networks.

The description of a given part must take into account the range of possible viewpoints. For example, an appendage that sticks straight out from the front of an object will appear longesι when the object is viewed from the side. As the viewpoint changes in moving around the object, the apparent length of such an appendage will change. When the object is viewed from the back, the appendage may not even be visible. An AM's description of such an appendage must take th.se different appearances into account; it must include (among other things) a specification of the range of lengths the appendage can appear to have. Spatial relations among parts may also vary with perspective. For example, an appendage may appear to jut out from the left side of an object from one perspective and to jut out from the right side from another perspective. These variations must be captured in the model. Other perspective-dependent relations may also hold. For example, the size of an appendage should be consistent with the apparent size of the rest of the object; if the length of an appendage suggests an object is being viewed from the side, the sizes of other object parts should be consistent with such a viewpoint.

Consider the simple object shown in Figure 7-A. Its silhouette will take the form of Figure 7-B when

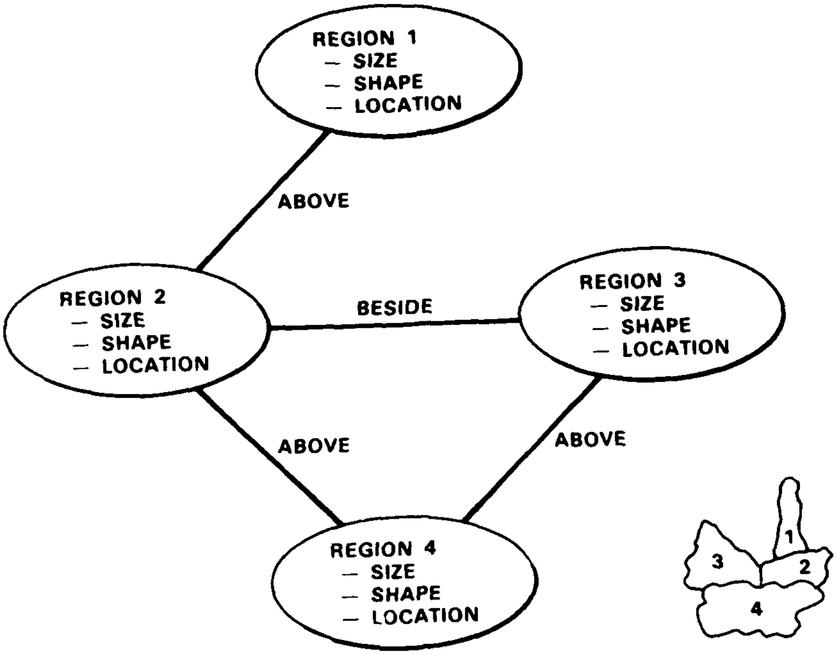
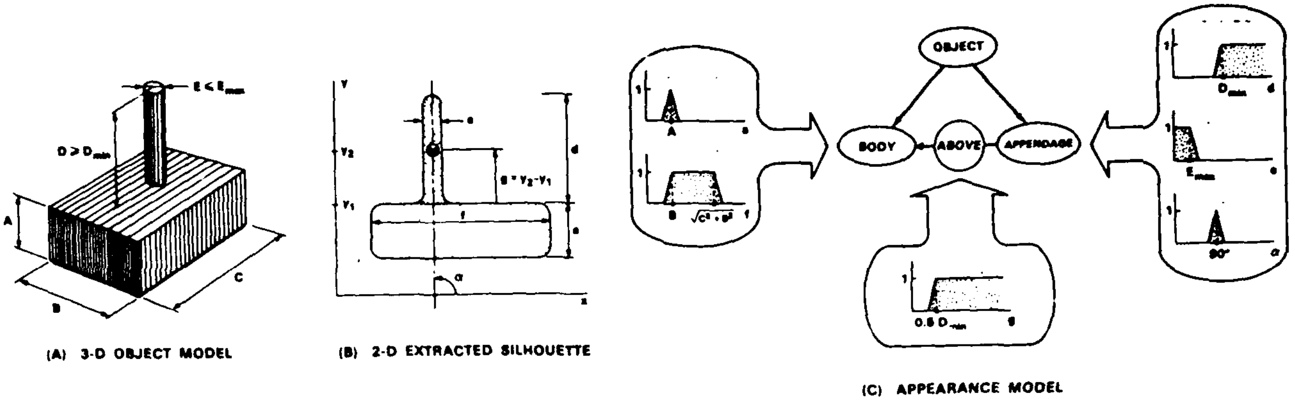


Figure 6: A notional attributed relational graph (ARG) for representing region information extracted from a range image

it is viewed from any one of a number of directions. Figure 7-C depicts an AM which describes the expected appearances of this object. This model is appropriate only for those types of imagery that lend themselves to the extraction of object silhouettes. The model assumes that the object will be viewed only from points at roughly the same elevation as the center of the object, and that the object is roughly horizontal in the image. For these viewpoints, the silhouette region of Figure 7-B should always consist of two parts. The corresponding appearance model is a part hierarchy in which an OBJECT is defined to be comprised of the two parts, BODY and APPENDAGE. Each of these parts is described by a set of *property functions* collectively termed a *property set.* A property function is a fuzzy predicate [Ohta 85] defined over the values of some region attribute(s). The value returned by a property function P for a given attribute a is a value in the closed range $[0,1]$ indicating the degree to which the attribute a satisfies the property defined by P.

For example, consider the property function defined over the measured quantity f determining the expected length of the part BODY. As shown in Figure 8, this function returns a value of 1.0 for regions whose length is between B and $\sqrt{B^2+C^2}$, and a value between 0.0 and 1.0 for regions whose length lies just outside this interval, and 0.0 for regions of any other length. The definition of this function reflects the fact that the projected length of the body is expected to vary between B and $\sqrt{B^2+C^2}$ for the assumed range of viewpoints. The rising and falling ramps on either side of this expected length interval are meant to allow for variations in measured region length due to sources of error such as image noise, inaccurate silhouette extraction, and inaccurate computation of region length. In a similar fashion, property functions are also defined for body height, appendage diameter, appendage height, and appendage major-axis orientation.

In addition to descriptions of the object's body and appendage, the model includes a constraint ABOVE on the spatial relationship between the two parts. This constraint is defined by a fuzzy predicate defined over values of g, which is the signed distance between the appendage region's apparent center-



(A) 3-D OBJECT MODEL

(B) 2-D EXTRACTED SILHOUETTE

(C) APPEARANCE MODEL

Figure 7: A simple object, its silhouette, and its appearance model

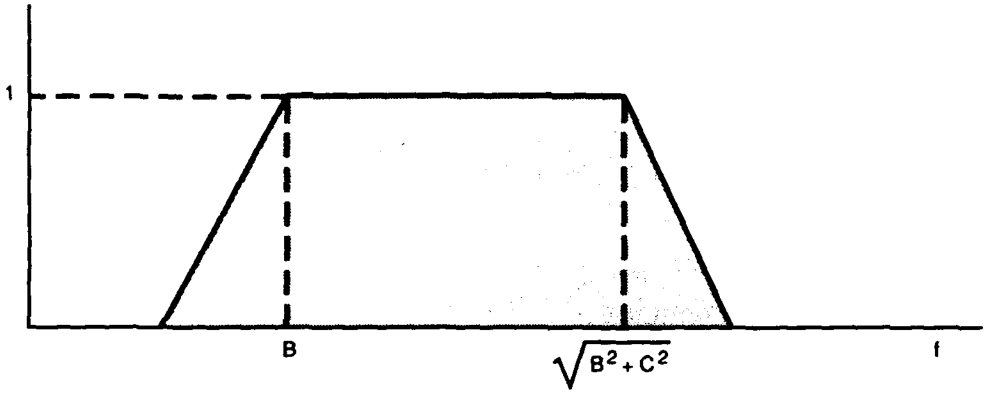


Figure 8: The property function for the expected length of the part BODY

of-gravity and the top of the body region.

In the present system, each AM is defined as a *part hierarchy*. In every AM, the root node represents an object and the children of each node represent the parts which comprise the object represented by that node. Parts may be defined in terms of their own parts, and so on; a part hierarchy may therefore extend to several levels. The terminal nodes of this hierarchy (the "atomic" parts) contain descriptions that can be matched against image-derived primitives to determine whether the atomic parts are present in an image.

Within a non-terminal AM node, *constraints on the relationships* between the node's parts may also be defined. These constraints typically include expected spatial relationships, the expected relative sizes of parts, and the ways in which parts are expected to appear joined to one another in the image. The constraints, together with the part hierarchy, define the expected possible appearances of an object in an image.

6 AM Hierarchies

The system's complement of AMs is organized into a hierarchy that comprises the system's knowledge about the kinds of objects it will be able to identify. The AM hierarchy acts as the model library for XTRS. It consists of a *specialization hierarchy* and the AMs themselves. The specialization hierarchy is a hierarchy of categories; the children of a given node represent subcategories of the category represented by that node. As discussed in the previous subsection, the AMs in an AM hierarchy describe objects in terms of their parts. To allow matching against these descriptions, an object characterization derived from the imagery must include a decomposition into primitive features. Since the kinds of image event and primitive descriptions derived in the contour-based approach differ from those derived in the region-based approach and since AM hierarchies are defined in terms of these descriptions, independent contour-based and region-based AM hierarchies are used to model targets of interest in XTRS-C and XTRS-R respectively.

Figure 9 depicts the AM hierarchy used in XTRS-C to describe the expected appearance of silhouette contours in the range imagery for three types of vehicles (V) – tank(T), howitzer(H), and amored personnel carrier (A).

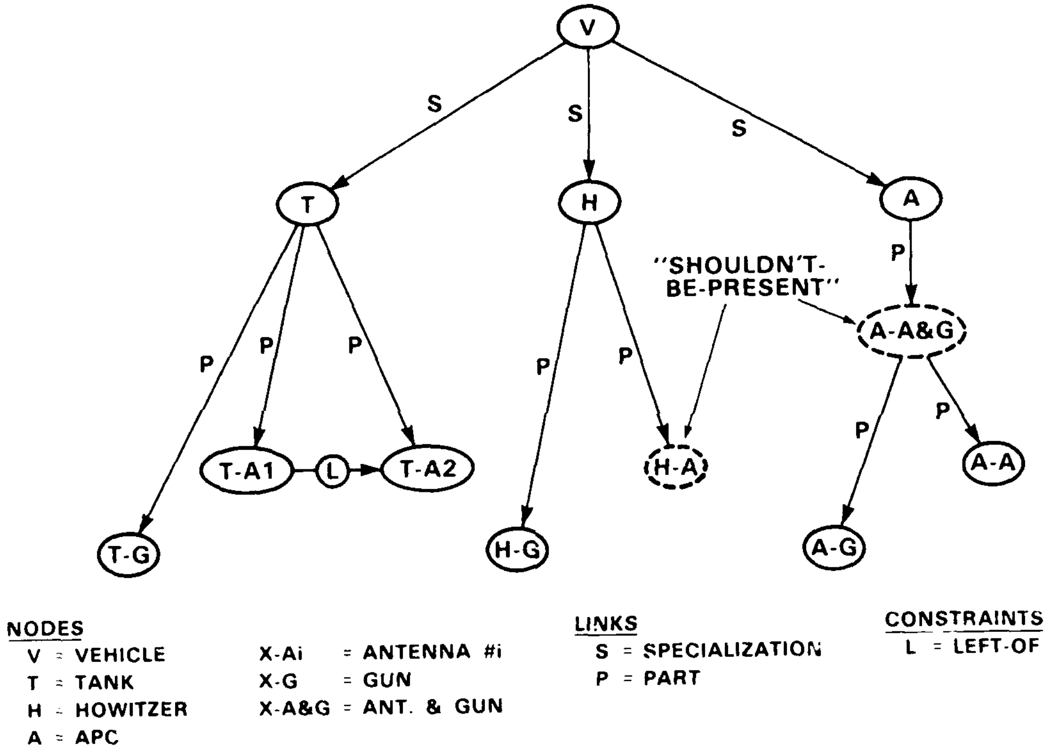


Figure 9: The appearance model hierarchy for the contour-based XTRX-C

The problem considered in XTRS-C is that of recognizing the type (A, H, T, or none of these) of a vehicle present in a range image solely on the basis of the presence or absence of characteristic appendages, such as gun barrels and antennas. These appendage-like parts are used here because they can easily be extracted using contour-based methods and because they are very useful in distinguishing the vehicles in the primary data base.

For category nodes such as T, H, and A, the definitions of the property sets relate to the characterization contained in the symbolic image-event description discussed in Section 4.1. The property sets of the terminal nodes relate to the symbolic primitive descriptions that contain attributes such as length and orientation. Composite nodes (that is, nodes that have subparts) do not possess a property set. Finally, constraints between nodes in the AM hierarchy express relationships between parts.

In the current version of the contour-based hierarchy the nodes V, T, H, and A do not include any property set, making the specialization hierarchy extremely simple. Each of the object nodes T, H, and

A is the root of an AM. For instance, the AM for the object T is comprised of the object node T; the three part nodes T-G, T-A1, and T-A2; and the constraint L. (This constraint is actually defined in the node T, but is shown separately here in order to indicate its presence). Each of the three nodes representing a part contains a property set which describes the ranges of shapes, sizes, and orientations expected for contour primitives corresponding to the silhouette of that particular part. The constraint L specifies that the first antenna T-A1 of object T must be to the "left-of" the second antenna T-A2. This example also illustrates the use of a "shouldn't-be-present" switch. For example, it is used to indicate that a vehicle with an antenna-like appendage cannot be of type H, at least in the primary data set.

The AM hierarchy for XTRS-R is shown in Figure 10. The property sets for nodes and the constraints of composite nodes are defined in terms of the region characterizations stored in the symbolic descriptions discussed in Section 4.2. Thus, the nodes of an AM describe objects in terms of region attributes such as length, width, and orientation, and the constraints of composite nodes are defined in terms of relationships among subregions. In this particular hierarchy, the property sets defined within each of the V, T, H, and A category nodes contain, among other things, the expected range of silhouette areas (in square meters) for objects belonging to each particular category.

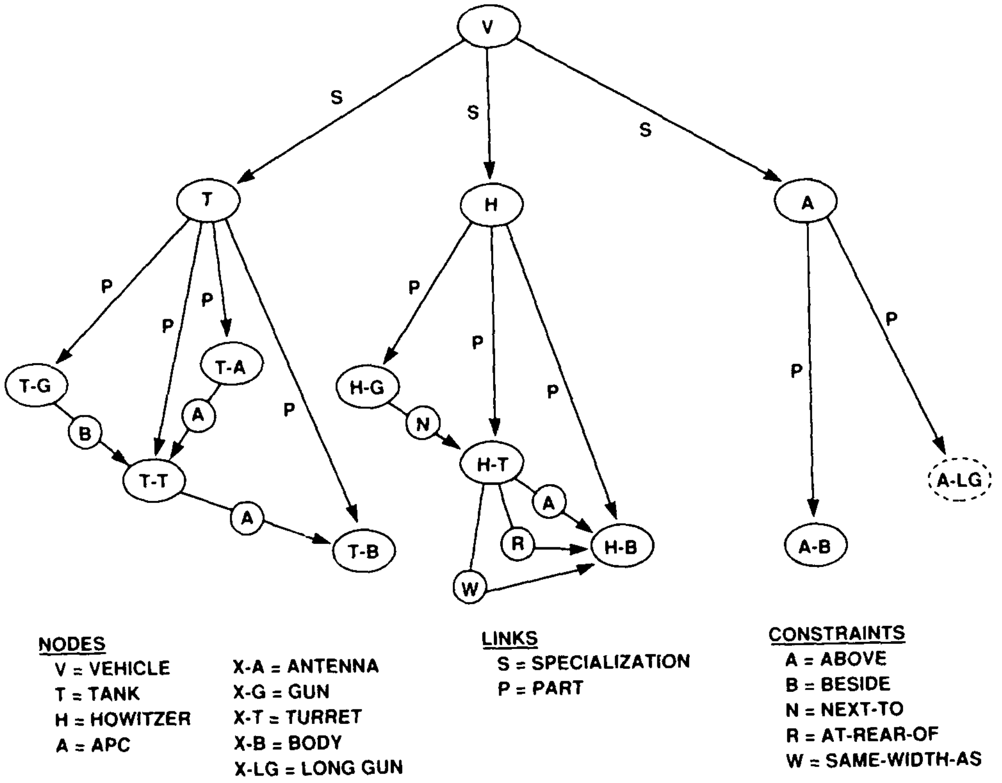


Figure 10: The appearance model hierarchy for the region-based XTRS-R

As in the case of the contour-based AM hierarchies, each of these nodes is the root of an AM. For instance, consider the AM for object T. Each of its four "part" nodes contains a property set which describes the variety of shapes and orientations expected for the region primitive that corresponds to the silhouette of that particular part. The constraints labeled A specify that two parts must be joined to one another with one part being "above" the other. For instance, the part T-A must be joined to and above the part T-T. The relationship "above" is defined in terms of a spatial relationship between two regions. Similarly, the constraint labeled "B" specifies that the part labeled T-G must be joined to and "beside" (i.e., left of or right of) the part labeled T-T.

7 Matching

Conceptually, the act of recognition in this narrow context consists of determining which, if any, of the AMs match the object characterizations extracted from the imagery and stored in symbolic descriptions. (A full description of the matching algorithm and the procedures that implement it is contained in [Verly 89a].) The recognition procedure implemented in the matching subsystem is intended to be general purpose. For some other application the object characterization subsystem and the content of the appearance model hierarchy may be very different, but it should be possible to use the same matching subsystem.

The inputs to the matching subsytem are a symbolic description representing the image event of interest (contour or region), the corresponding symbolic primitive descriptions, and an AM hierarchy. The output is the identity of the object in the AM hierarchy that best matches the input symbolic descriptions extracted from the imagery. The processing done by the matching subsystem consists of three steps: *symbolic matching*, *belief computation*, and *decision*. The first step (symbolic matching) consists of two phases, referred to as *pruning* and *AM matching*. The overall organization of the system is depicted in Figure 11.

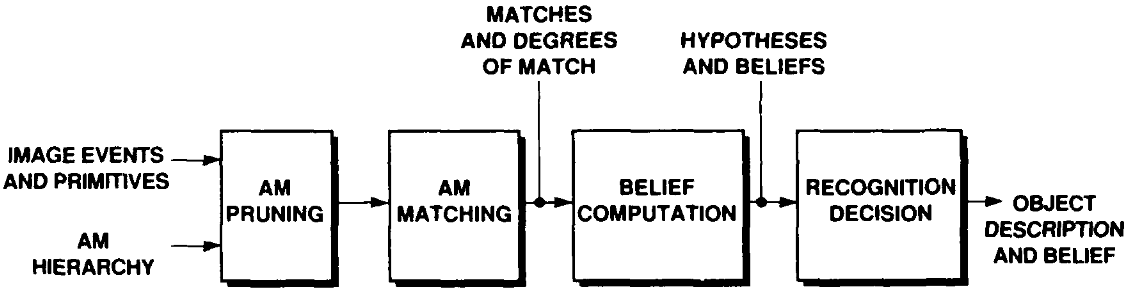


Figure 11: Processing steps in the matching subsystem

The first phase of symbolic matching, called the *pruning* phase, attempts to eliminate whole categories of AMs from further consideration. This is accomplished by matching the image-event description in a depth-first manner against the property sets of the nodes in the specialization hierarchy. On reaching a category node that fails to match the image-event description, the portion of the AM hierarchy rooted at that node may be "pruned" (i.e., removed from the hierarchy) since the image-element cannot represent a match to any of that node's subcategories. In this way, whole classes of AMs can be excluded as possible matches to the image-event description, reducing the number of AM's that must be considered in the second phase. The AMs that remain as possible matches after the pruning phase are called *active AMs*.

In the second phase of symbolic matching, called the *AM-matching* phase, the primitive descriptions are matched against the part descriptions and constraints of the active AMs. This matching is carried out independently for each AM using a depth-first matching procedure. This procedure matches the input primitive descriptions against each of the terminal part nodes of the AM and evaluates the constraints defined among the parts of composite nodes. A *degree of match* is computed for each match of a primitive description to the property set of a terminal node and for the evaluation of each constraint with respect to a given set of primitive descriptions. (Each element in such a set of primitive descriptions represents a match for one of the parts involved in the constraint). For composite nodes, a degree of match is computed for each set of primitive descriptions that represents a possible match for the parts of the composite node. This is done by combining the degrees of match for that node's parts and constraints given the set of primitive descriptions. When AM matching is completed, each node of each active AM will contain a list of possible matches (to the node's property set for terminal nodes, or to the node's parts for composite nodes) in which each element has an associated degree of match. A simple example of the symbolic matching step is discussed below, but the details are complex and cannot be covered in this short article. The reader is referred to [Verly 89a] for more information on this subject.

As illustrated in Figure 12, a silhouette region extracted from the laser radar imagery is characterized

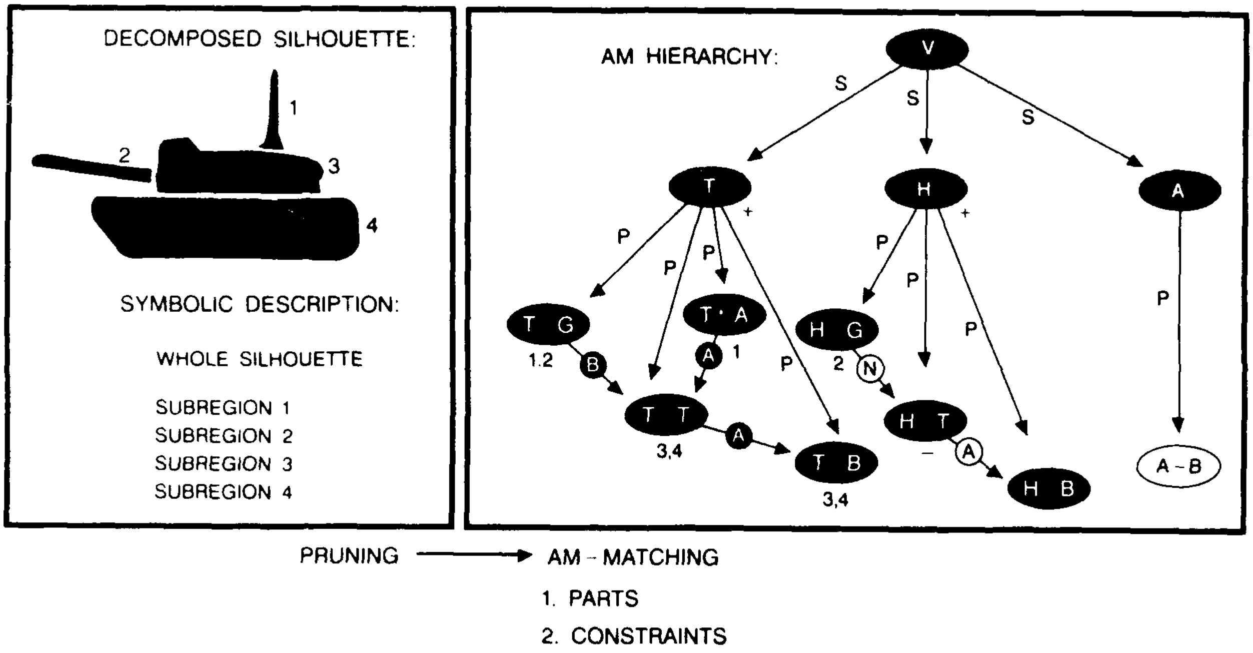


Figure 12: Symbolic matching consists of the pruning phase and the AM matching phase. AM matching embodies both part matching and constraint evaluation.

by its size and other attributes. It is also decomposed into subregions that are characterized by size, shape, orientation, location, and their spatial relationships to one another.

The matching of the object characterization to the appearance model (AM) hierarchy begins with the *pruning* phase. Pruning attempts to remove from further consideration those object hypotheses that do not match well with the gross characteristics of the extracted silhouette. We begin with the root node of the AM hierarchy, labelled V for vehicle. Since the size of the extracted silhouette is consistent with the property set stored in the V node, we drop to the next level of detail, which in this case contains the object hypothesis nodes themselves, T for tank, H for howitzer, and A for armored personnel carrier.

The silhouette characterization is compared to the model information stored in the T, H, and A nodes. In this case, the gross silhouette satisfies the property sets in the nodes T and and H but not the property set of the A node. Thus T and H become the active hypotheses; A has been pruned from the tree of hypotheses and is not considered further for this silhouette.

In the second phase, *AM matching*, the four subregions resulting from the silhouette decomposition are matched to the various parts contained in the AMs of the active hypotheses. The tank AM, for example, contains the parts gun $(T-G)$, antenna $(T-A)$, turret $(T-T)$, and body $(T-B)$, as well as the spatial relationships between the parts (A - above, B - beside, N - next to). Using the property sets of the various parts, we can match subregion 1 to the tank gun (Subregion 1 could correspond to the gun if it were greatly elevated and oriented toward or away from the sensor.) or the tank antenna, subregion 2 to the tank gun or the howitzer gun, subregion 3 to the tank turret or tank body, and subregion 4 to the tank turret or tank body. (Subregions 3 and 4 do not satisfy the properties for the howitzer turret or body.) Since the $T-T$ must be above the $T-B$, the assignment of subregion 3 to the turret and the assignment of subregion 4 to the body are consistent with this constraint. Finally, matching subregion 1 to part $T-A$ and subregion 2 to part $T-G$ gives a higher degree of match for the tank hypothesis than the alternative of matching subregion 1 to $T-G$ and leaving subregion 2 unmatched. For the tank hypothesis then, the assignment of subregions to model parts that gives the best possible matchis:

$$1 \rightarrow T-A,\ 2 \rightarrow T-G,\ 3 \rightarrow T-T,\ 4 \rightarrow T-B$$

For the howitzer hypothesis, the best possible match is the assignment of subregion 2 to the howitzer gun; no other subregions satisfy the properties of the other howitzer parts.

In the *belief-computation* step, degrees of match computed during AM matching are used as support for the competing object hypotheses. An object hypothesis is either a hypothesis of the form "The image event under consideration corresponds to the object o," where o is one of the object hypotheses of the AM hierarchy, or the null hypothesis "The image event being considered does not correspond to any of the objects described in the AM hierarchy." For each active AM, there will be a combination of primitive descriptions which comprises the *best* match to that AM. Each of the primitive descriptions in this best match will be matched to one of the terminal parts of the AM. The degrees of match for the highest level parts and constraints in each active AM (given this best combination of primitive descriptions) are treated as degrees of support for the corresponding object hypothesis. The method provided by the Dempster-Shafer theory of evidence [Shafer 76,Gordon 85,Verly 89b] is used to represent and combine this support. In the Dempster-Shafer formalism, a mapping termed a *basic probability assignment (bpa)* is used to assign degrees of support to competing hypotheses on the basis of individual items of evidence. In the present system, the degrees of match computed during AM matching for the best match to each active AM are used to define bpas expressing support for that AM's corresponding object hypothesis. All such bpas defined for active object hypotheses are then combined using Dempster's combination rule to produce a single bpa. This bpa, which is the output of the belief computation step, quantifies the support assigned to the various object hypotheses as a result of matching the image-event description and primitive descriptions against the AM hierarchy.

At this point we have a degree of belief for each active hypothesis. (The degree of belief for hypotheses that were rejected in the pruning stage is necessarily zero.) A decision rule is needed to translate this set of beliefs into a recognition decision. XTRS uses a very simple rule: the hypothesis with the largest degree of belief is chosen. However, in a more sophisticated system a more complex decision rule might be used. For example, it may make sense to choose the hypothesis with the largest degree of belief only if it exceeds its competitors by a wide margin. If there are two hypotheses whose degrees of belief are almost equal, then it may be prudent to select the compound hypothesis that it may be either possibility and call for additional processing to distinguish among them.

8 Control

The primary responsibility of the control subsystem is to provide flexibility in solving a recognition problem. Ultimately the control system should embody a variety of recognition strategies along with the means of selecting an appropriate strategy for a given situation. This topic is one of our current research interests; there is a great deal more work to be done here. We have begun by investigating the use of high-level feedback to improve our recognition results.

A common means of achieving flexibility is through the use of thresholds and tunable coefficients that adapt a specific function to a specific image condition or object class. For example, the tolerance used for creating a polygonal approximation of silhouette contours varies with the range to the target. We have implemented this kind of flexibility by developing a general procedure which is supplied with a set of rules describing properties of the object class being sought. However, selection of the rules requires knowledge of the viewing conditions or of the object being sought.

In our approach then, the development of a control subsystem rests on the machine intelligence techniques of knowledge management and problem solving. In the area of automated target recognition, rule-based techniques can be used to select coefficients and algorithms on the basis of image context. For example, one approach has been to use knowledge of the terrain and map data to select scene areas as likely sites for target position [McKeown 83a,McKeown 83b]. An alternative approach evaluates such object parameters as size, contrast, motion, shape and color to establish context and select the most appropriate processing mode [Wootten 88]. Instead of attempting to characterize image context prior to scene processing, XTRS includes a rule-based mechanism for feedback.

For our purposes, feedback consists of an evaluation of results (intermediate or final), which is the basis for deciding what parameters to adjust and where to direct the flow of control. Feedback can be initiated for a variety of reasons. The most important are as follows:

Algorithm failure: A critical intermediate result is missing.

Dead end detection: An intermediate result has features that contradict the object class being sought. Consequently, there is no need to finish processing.

Selected target is unknown: Examine primitives for clues indicating poor extraction or inappropriate decomposition. If found, adjust parameters and repeat processing.

Belief for a selected target is indecisive: Once again, check for clues of inappropriate processing.

Figure 13 shows the relationship of the three main processing modules and 4 distinct feedback modules. Each main processing module has an associated local feedback module that can initiate short-loop feedback. These feedback modules can be used to correct algorithm failures. For example, the failure to find a reasonable polygonal approximation usually results in an increase of the tolerance parameter, and a repeated attempt to find a reasonable polygonal approximation. Alternatively, intermediate results may indicate that the system is working on a dead end: a silhouette output of the extraction module that cannot reasonably be a military vehicle causes the rejection of that region. Control returns to the beginning of the extraction module, where the next likely target region is extracted. If no problems are detected, flow of control is passed on to the next processing module. Local feedback modules also have the option of deciding that the most appropriate response is outside of the module it is associated with. In such a case, flow of control is passed to the global feedback module. Here, the decision is made whether to quit altogether, to return to the extraction module for another target region, or if the current region is judged salvagable, to change some parameter and return to any of the three processing modules.

With any closed loop system, infinite loops are a distinct possibility. As a fail-safe mechanism, each feedback module is equipped with a counter, which is incremented each time that feedback module issues an instruction, and a pass limit. If a module's counter exceeds its associated pass limit, the module simply quits issuing feedback instructions. The maximum number of feedback instructions equals the sum of the pass limits of the 4 feedback modules, thereby ensuring termination.

Although the complete feedback system has been implemented, we are still gaining experience with how to use it effectively.

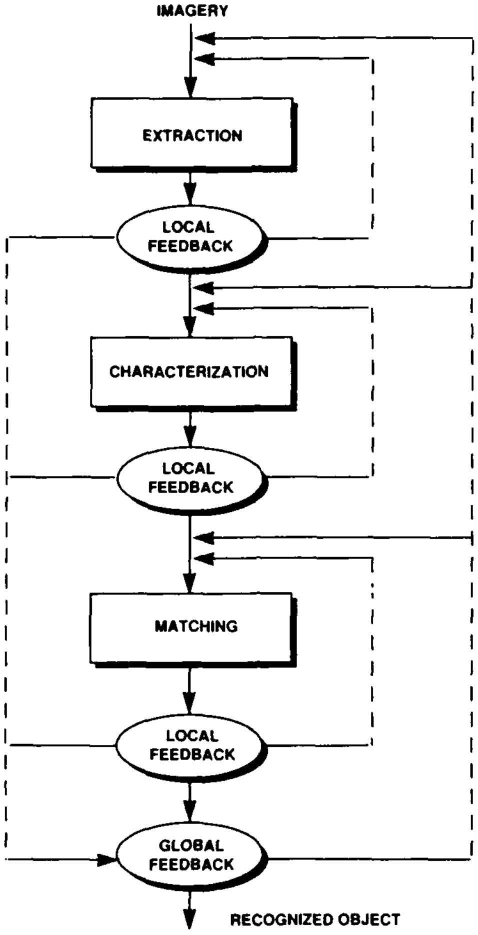


Figure 13: Control Structure. Arrows indicate direction of the flow of information.

9 Results

Because of the difficulty and expense of printing color photographs, we are not able to show examples of XTRS in operation. Individual examples of automatic target recognition will be shown as part of the oral presentation of this paper. The entire processing sequence from the input images to recognized and labeled objects is executed automatically by the system using a single set of processing parameters, and contour- and region-based processing are completely independent.

Example 1 is a tank at an oblique angle to the sensor. Both XTRS-C and XTRS-R correctly recognized this object despite failing to extract a perfect silhouette. Because of the noise in the range image, XTRS-C did not extract all of the gun barrel and XTRS-R failed to extract one of the antennas. The other antenna was not recognized by XTRS-R because it was longer than allowed by the tank appearance model. Despite these small problems, both systems acquired enough information to make a correct decision. This example also suggests that an intelligent combination of the XTRS-C and XTRS-R results might lead to a correct labelling of all parts.

Example 2 is an APC at right angles to the sensor line of sight. Both systems correctly identified the vehicle based primarily on its size and lack of other distinguishing features.

Example 3 is a howitzer facing the sensor. The gun is elevated enough so as to be visible above the howitzer's body. Both systems correctly recognized the vehicle and correctly labelled the gun inspite of this unusual geometry. The appearance models for both systems contained enough information to indicate that the gun could appear in this way. The gun is thicker than an antenna so it was not mistaken for one.

The present system has been exercised on 40 scenes similar to Examples 1-3. These scenes are views from various aspects of vehicles of the types T, H, and A and for various configurations of those which are articulated (types T and H). XTRS-C and XTRS-R have each attained an overall recognition rate of 100% on this limited data set, as indicated in Figure 14.

TARGETS AT 700 m

		TANK (21)	HOWITZER (15)	APC (4)
RECOGNIZED AS	T	100%	0%	0%
	H	0%	100%	0%
	A	0%	0%	100%
	NONE	0%	0%	0%

Figure 14: A confusion matrix showing how often tanks, howitzers, and APC were correctly recognized and how often they were confused with one another. The targets were at a 700m range with a high-contrast background.

One of the consistent difficulties with constructing vision systems is ensuring robust behavior for a wide range of viewing conditions. The reader should interpret 100% recognition in light of the fact that the 40 test images were relatively easy to handle. In these images, the objects of interest were isolated and filled a large portion of the frame, noise was manageable and object-background contrast was good. In practice, images with excessive noise, distant objects, occlusions, and poor contrast between object and background will be encountered. Adaptive control and goal-driven (top-down) exploration techniques will be required to handle these more difficult cases.

Examples 4 and 5 are more difficult cases handled sucessfully by XTRS-C and XTRS-R. Example 4 is a howitzer perpendicular to the sensor line of sight at a range of about one kilometer. Because of the depression angle at which this vehicle was viewed, the background is relatively close to the vehicle causing contrast in the range image to be reduced, making region extraction and characterization more difficult. Both systems are able to extract appropriate silhouettes and correctly recognize the vehicle. Figure 15 shows a confusion matrix for tanks, howitzers and APCs of various orientations in similar imagery.

TARGETS AT 1000 m

		TANK (21)	HOWITZER (16)	APC (4)
RECOGNIZED AS	T	91%	7%	25%
	H	0%	93%	0%
	A	9%	0%	75%
	NONE	0%	0%	0%

Figure 15: A confusion matrix showing how often tanks, howitzers, and APC were correctly recognized and how often they were confused with one another. The targets were at a 1000m range with a low-contrast background.

Example 5 is a target in clutter at a range of about 1.3 kilometers. In addition to the vehicle, there are poles and general foliage clutter. XTRS-R was able to extract the correct region and identify it as a tank based on the size of the turret and body subregions. This particular example was sucessful, but in general XTRS has difficulty segmenting out the correct target region in imagery this complicated. There is clearly room for further work in this area.

One way of acquiring a data set with more objects at a greater variety of viewing angles is to generate images synthetically. We have developed a synthetic image generation system that uses 3-D object information to compose a range image and an angle-of-incidence image. These two images are then degraded by simple noise models to obtain realistic range and intensity images. This capability allows us to generate data sets to study the performance of XTRS as a function of variables such as range to target. There are seventeen target models that are currently available to the synthetic image generator.

Using the synthetic data sets, we can perform experiments to evaluate the performance of XTRS quantitatively. Figure 16 shows a graph of the recognition performance of XTRS-C and XTRS-R as the range to the vehicle increases out to three kilometers. (The angular resolution is the same as that found in the real imagery.) Each data point represents twenty trials using eight tanks, eight howitzers and

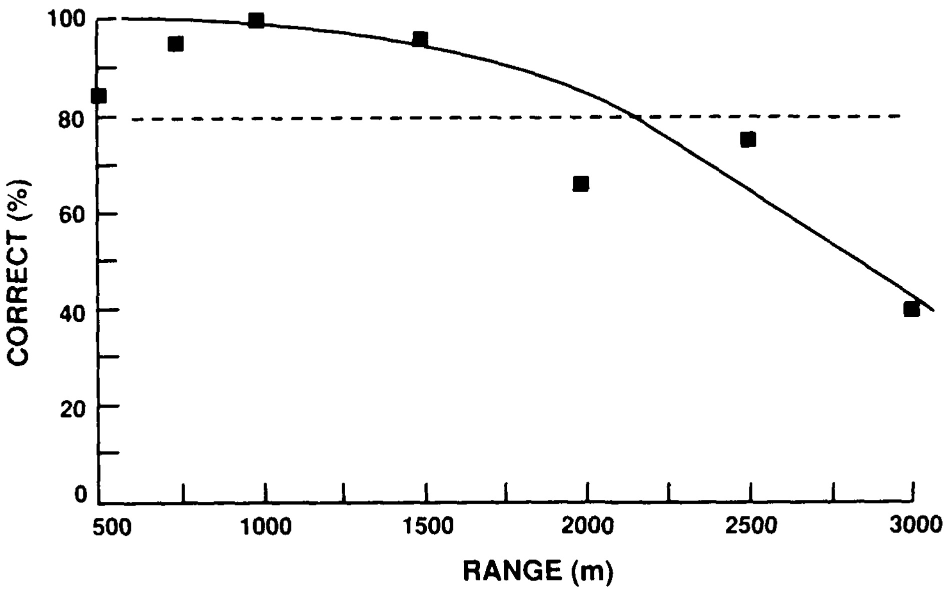


Figure 16: Recognition performance as a function of range tc target.

four APCs at various orientations and articulations as the target. (This same mix of synthetic targets is used in all the experiments described below.) Even out to three kilometers there are enough pixels across the target to recognize its shape.

XTRS can also be used to investigate the effects of altering some of the basic sensor characteristics. For example we are interested in learning how XTRS performance degrades with increasing levels of noise and how it improves with better angular and range resolution. Once again we can use the synthetic image generator to compile appropriate data sets to begin addressing these questions. For example, Figure 17 shows how performance degrades as the percentage of outliers in the range image is increased. (The percentage of outliers is related to the laser carrier- to-noise ratio. The real imagery has an outlier percentage of about 1%.)

A sensor with increased range accuracy will give images with better contrast between target and background and will begin to convey some measure of the target's third spatial dimension. This becomes important in the airborne ATR application where increasing depression angle reduces the range discontinuity between the top of an object and the background above it. Figure 18 shows how the recognition performance of XTRS-C varies with depression angle for range accuracies of two meters and six meters. Figure 19 contains a similar graph for XTRS-R. Higher range accuracy can improve recognition for the low-range-contrast situations that will be found in the airborne target recognition application.

10 The Future

In this paper we have discussed our experimental target recognition system. In XTRS the model library takes the form of an appearance model hierarchy. The input to the matching subsystem consists of an AM hierarchy and the set of image event characterizations derived from the imagery. Each characterization consists of a symbolic and numeric description of a single image event (in our case, either a contour or a region) extracted from the imagery and descriptions of the primitive features resulting from

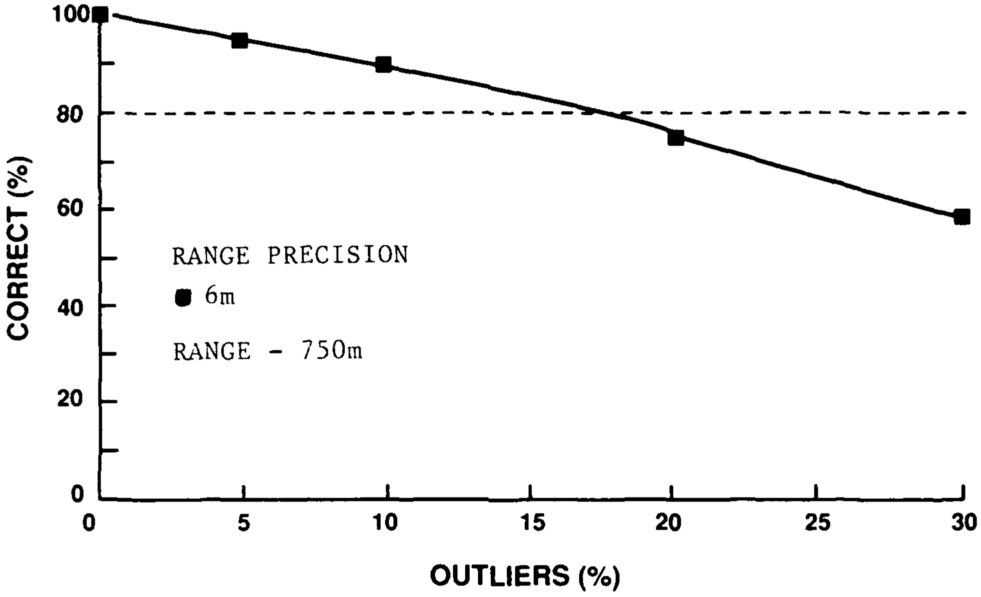


Figure 17: Recognition performance as a function of noisy pixels

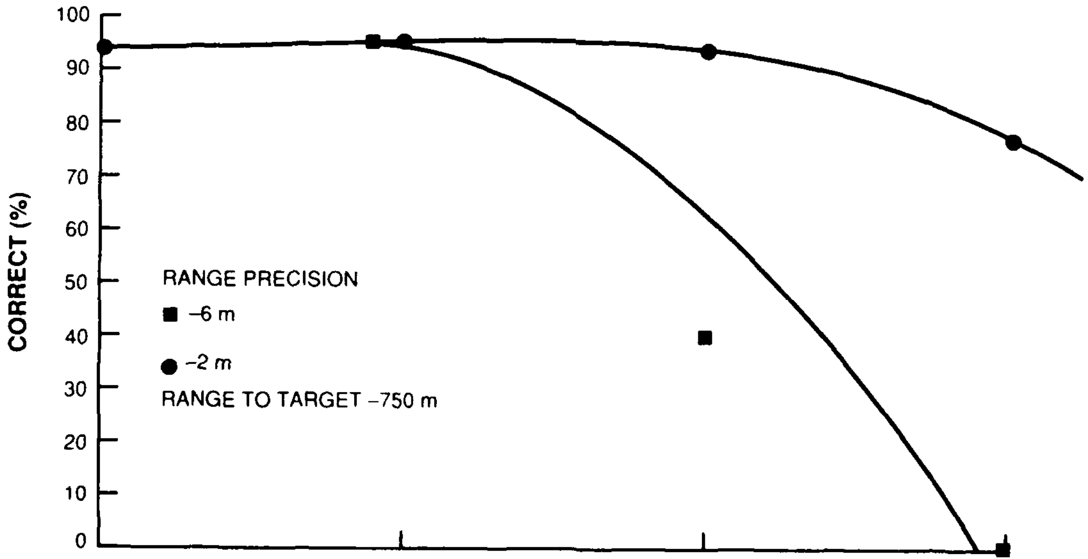


Figure 18: Recognition performance for XTRS-C as a function of depression angle with range accuracy as a parameter.

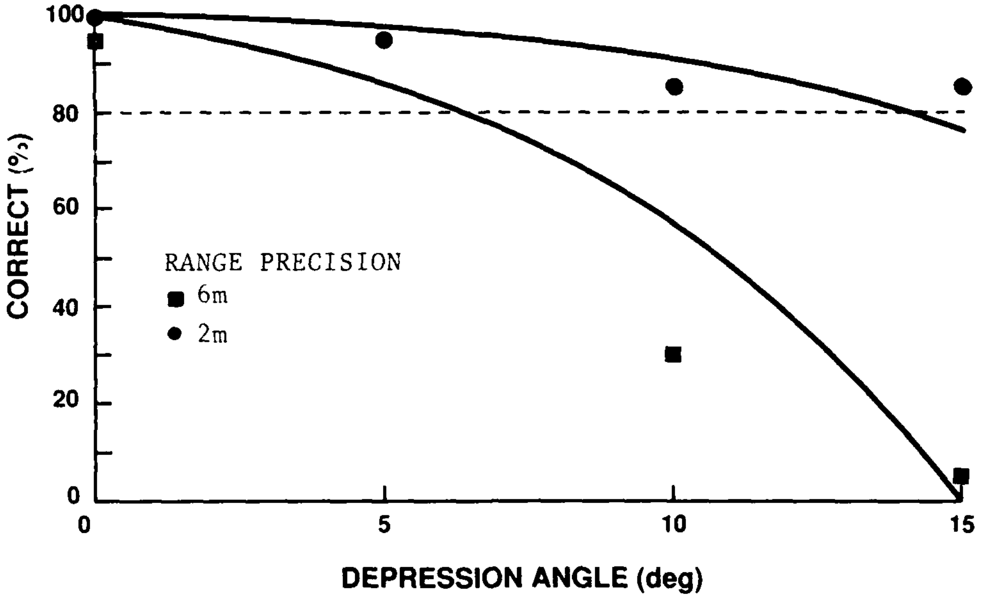


Figure 19: Recognition performance for XTRS-R as a function of depression angle with range accuracy as a parameter.

the decomposition of the extracted event. The procedure that matches the image-derived descriptions against the AM hierarchy is application-independent. The only system components that change from one application to another are the AM hierarchy containing the knowledge about the objects of interest and the algorithms necessary to extract desired image events and to dccompose them into primitives. Thus, our approach has a wide range of potential applications.

XTRS attempts to recognize vehicles using range silhouettes extracted from coarse-resolution range imagery. The important steps of silhouette extraction, decomposition, and recognition were discussed in detail both in the contour- and region-based cases. In both cases, the system operates in a fully autonomous fashion from raw data to object recognition and has been observed to perform extremely well for images similar to those of Examples 1-3. We have demonstrated that the concept of AMs evaluated in an evidential formalism can achieve good object recognition of articulated, imprecisely defined objects present in real laser radar images with significant noise and coarse range resolution.

There is room for improvement in all the constituent subsystems of the generic model-based recognition system (Figure 1). We can expect improvements in the detection, extraction, and characterization of events in the imagery from two sources. New sensors may offer new features and new discriminants that may be exploited by new characterization algorithms, and new algorithms may be developed that can combine measurements from different sensors at a very low level. For example, we have used the pixel-registered intensity and range measurements together in the region-based system to extract target regions. A more sophisticated region extraction algorithm may use several bands of optical or infrared imagery in conjuction with laser range and intensity images.

The realization of the model library is crucially important. Robust recognition systems must use a variety of clues to deduce the identity of an unknown object. That requires using diverse pieces of information at the right time in the matching process. The information represented in the model library must be easily accessible to the matching subsystem to keep the computational cost reasonable.

Building the individual components of a large ATR system may draw on powerful mathematical and theoretical concepts, but assembling these components into a system is currently more an art than a science. There does not exist any global theoretical framework for building complete vision systems. The only guiding principle adopted by virtually all systems is the division of the problem into three stages, often referred to as low-level vision (silhouette extraction in XTRS), intermediate-level vision (silhouette decomposition), and high-level vision (silhouette recognition). However, within each level there are a variety of well developed techniques that can be applied to particular aspects of the recognition problem.

Our approach to the design of XTRS has been to use established and mathematically well-understood techniques wherever feasible (*e.g.*, mathematical morphology, difference of gaussian filtering, theory of evidence). *Ad hoc* techniques have been used where necessary, most notably in the silhouette decomposition step. When recourse to *ad hoc* processing has been necessary, a special effort was always made to create a modular building block, whose operation can be explained in lucid terms, whose behavior is predictable, and whose domain of applicability can be readily extended. (Examples of *ad hoc* building blocks satisfying these requirements are the contour and region decomposition steps.)

The AM hierarchy evaluation system also illustrates our attempt to impose modularity and rigor: This system is application independent and the calculation of degrees of match has a sound mathematical foundation. The use of modular, well-understood building block in XTRS allows us to quickly reconfigure the system and to replace any algorithm with an alternate processing scheme.

Progress on the general ATR problem has been slow. The performance of an object recognition system is difficult to analyze mathematically; we are hindered by the lack of a global theoretical framework. It is also difficult to make progress experimentally because of the large amount of imagery that must be processed to investigate ATR performance under many conditions. Fortunately, technology continues to provide us with cheaper and more powerful computational resources, and computer science is beginning to provide mechanisms for the rapid conversion of ideas into executing algorithms. Our ability to experiment is improving.

In our work we have tried to address the whole problem. Our goal from the beginning was to build an end-to-end system, one that takes sensor images as its input and makes a recognition decision as its output. Having an end-to-end system allows us to experiment with it, augment it, reconfigure it, and use it to discover and evaluate new approaches to the ATR problem. In a real system, performance improvements may arise from a synergistic combination of suboptimal processing algorithms. Conversely, the replacement of a particular algorithm by a more nearly optimal, but computationally more expensive, alternative may not improve the performance of the end-to-end system significantly. Our experimental system provides an excellent testbed for observing and understanding such effects.

References

[Aull 88] A. M. Aull and R. A. Gabel, "Machine Intelligence Applied to Radar Object Modeling," *MIT Lincoln Laboratory Technical Report*, TR-818, October 12, 1988.

[Delanoy 89] Richard L. Delanoy, Jacques G. Verly, and Bryan D. Williams, "Region-Based Object Recognition Using Appearance Models," Lincoln Laboratory Technical Report, in preparation.

[Eshera 86] M. A. Eshera and King-Sun Fu, "An Image Understanding System Using Symbolic Representation and Inexact Graph Matching," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol. PAMI-8, no. 5, Sept. 1986.

[Esselman 87a] T.R. Esselman and J.G. Verly, "Some Applications of Mathematical Morphology to Range Imagery," *Proc. of 1987 IEEE Int. Conf. on Acoust., Speech, and Signal Processing (ICASSP)*, pp. 245-248, Dallas, TX, Apr. 6-9, 1987.

[Esselman 87b] T. R. Esselman and J.G. Verly, "Applications of Mathematical Morphology to Range Imagery," *MIT Lincoln Laboratory Technical Report*, TR-797, December 23, 1987.

[Gordon 85] J. Gordon and E. H. Shortliffe, "The Dempster-Shafer Theory of Evidence," in *Rule-Based Expert Systems* (B. G. Buchanan and E. H. Shortliffe, eds), Addison Wesley, Reading, MA, pp. 272-292, 1985.

[Gschwendtner 83] A. B. Gschwendtner, R. C. Harney, R. J. Hull, "Coherent IR Radar Technology," in *Optical and Laser Remote Sensing*, D. K. Killinger and A. Mooradian, eds. Series in Optical Sciences, Springer-Verlag, 1983.

[Kanade 87] T. Kanade, *Three-Dimensional Machine Vision*, Kluwer Academic Publishers, Boston, MA, 1987.

[Marr 82] D. Marr, *Vision*, W. H. Freeman and Company, San Francisco, CA, 1982.

[McKeown 83a] David M. McKeown, Jr., "MAPS: The organization of a spatial database system using imagery, terrain and map data," *Proceedings: DARPA Image Understanding Workshop*, pp. 105-127, June 1983.

[McKeown 83b] David M McKeown, Jr. and John McDermott, "Towards expert systems for photo interpretation," *IEEE Trends and Applications*, pp. 33-39, May, 1983.

[Ohta 85] Y. Ohta, *Knowledge-based Interpretation of Outdoor Natural Color Scenes*, Pitman Publishing, Boston, MA, 1985.

[Pavlidis 79] T. Pavlidis and F. Ali, "A Hierarchical Syntactic Shape Analyzer," IEEE Transactions on Pattern Analysis and Machine Intelligence, Vol. PAMI-1, No. 1, pp. 2-9, Jan. 1979.

[Ramer 72] U. Ramer, "An Iterative Procedure for the Polygonal Approximation of Plane Curves," Computer Graphics and Image Processing, Vol. 1, pp. 244-256, 1972.

[Selfridge 82] P. G. Selfridge, *Reasoning about Success and Failure in Aerial Image Understanding*, Ph. D. Thesis, University of Rochester, Rochester, NY, 1981.

[Serra 82] J. Serra, *Image Analysis and Mathematical Morphology*, Academic Press, London, 1982.

[Shafer 76] G. Shafer, *A Mathematical Theory of Evidence*, Princeton University Press, Princeton, NJ, 1976.

[Van Hove 87] P. L. Van Hove, "Model-Based Silhouette Recognition," *Proc. of the IEEE Computer Society Workshop on Computer Vision*, pp. 88-93, Miami Beach, FL, November-December 1987.

[Van Hove 88] P. L. Van Hove and R. A. Jaenicke, "Model-Based Silhouette Recognition," *MIT Lincoln Laboratory Technical Report*. TR-817, November 25, 1988.

[Verly 86] J.G. Verly, P.L. Van Hove, R.L. Walton, and D.E. Dudgeon, "Silhouette Understanding System," *Proc. of 1986 IEEE-IECEJ-ASJ Int. Conf. on Acoust., Speech, and Signal Processing (ICASSP)*, pp. 1457-1460, Tokyo, Japan, Apr. 8-11, 1986.

[Verly 87] J. G. Verly, B. D. Williams, and D. E. Dudgeon, "Automatic Object Recognition from Range Imagery Using Appearance Models," *Proc. of the IEEE Computer Society Workshop on Computer Vision*, pp. 244-246, Miami Beach, FL, November-December 1987.

[Verly 89a] Jacques G. Verly, Bryan D. Williams, and Richard L. Delanoy, "Three-Dimensional Object Recognition Using Appearance Models," Lincoln Laboratory Technical Report, in preparation.

[Verly 89b] Jacques G. Verly, "A Review of the Dempster-Shafer Theory of Evidence," Lincoln Laboratory Technical Report, in preparation.

[Wootten 88] John Wootten, Carl Carpenter, Gregory Hobson, and James Keller, "A multiple hypothesis rule-based automatic target recognizer," *Proceedings of 4th International Conference on Pattern Recognition*, Josef Kettler, ed., Springer-Verlag, Cambridge, England, U.K., pp. 315-324, March 1988.

UNIFICATION AND INTEGRATION OF VISUAL MODULES: An Extension of the Marr Paradigm

John Y. Aloimonos
Computer Vision Laboratory
Center for Automation Research
University of Maryland
College Park, MD 20742-3411

ABSTRACT

To address the problem of computational vision, a methodology must be followed. There is a general principle for designing large and complex information systems. First we divide the system into functional components which break the overall task into autonomous parts, and analyze these components. Then we must choose the representation of information within the subsystems and the language of communication among them. After this, the details of the systems are tested individually, in pairs and all together. At least the above principle should be followed in analyzing a complex information system, such as a visual system.

The Marr paradigm work has called for a deep study of the individual functional autonomous components, the visual modules. And although such research continues to address several important unexplored issues, it has become clear that most aspects of passive monocular vision (modules) are problematic in the sense that most modules are faced with the task of solving ill-posed problems.

A problem is ill-posed when its solution does not exist, is not unique or does not depend continuously on the data. In other words, most early vision modules either do not have a unique solution or, if one can show that they do, usually the solution is unstable in the presence of noise. Thus, we feel the time is ripe for moving to the next natural step (extension) in computational vision research according to the above-mentioned general principle, i.e. to work towards the integration of the modules, in order to achieve unique and robust reconstruction. However, before this step is attempted, we need to first study the stability of the existing modules in the presence of noise, i.e. to develop "optimal" modules—modules that optimally estimate what they are supposed to compute, in order to realize how good the best is—and second to examine the commonalities in the existing modules in order to understand what are the minimal hardware requirements.

In this paper I attempt to present a broad view of the computational vision field, i.e., to look at it from several points of view and different perspectives. For that I first needed to describe a classification of existing works in order to understand why we do what we do and what are some of the ultimate goals of computer vision research. Then I proceed to describe some examples from our research along the above-mentioned lines. In particular, in studying individual modules, the celebrated problem of structure from motion is examined from a stability and complexity point of view. After this, a theory of regularization effective in the presence of discontinuities is presented; the theory can be used for both unifying as well as integrating early vision modules. In other words, the same theory can be used for developing solutions to early vision modules such as edge detection, interpolation, shape from texture, patterns, shading, etc., lightness constancy, image compression, deblurring, geometric decomposition, model-based vision, and so on, as well as for integrating the abovementioned modules, i.e. combining information from different cues in order to facilitate robust reconstruction. In the course of the analysis it is shown how one can basically learn the constraints that relate 3-D characteristics to 2-D image cues. Finally, I describe our recent research in high-level vision and in particular some aspects of the problem of visual learning. This paper could be considered as a methodological summary of our forthcoming book[2] by Academic Press.

[1] The help of Barbara Burnett and Sandy German in preparing this paper is gratefully acknowledged.

[2] J. Aloimonos and D. Shulman, *Integration of Visual Modules: An Extension of the Marr Paradigm*, Academic Press, Boston, in press.

1. INTRODUCTION

There is a standard way to design large and complex information systems as research in computational fields has shown. First we divide the system into functional components which break the overall task into autonomous parts, and analyze these components. Then we must choose the representation of information within the subsystems and the language of communication among them. After this, the details of the systems are tested individually, in pairs, and all together [Feldman, 1985]. At least the above principle should be used in analyzing a complex information system, such as a visual system. With respect to research in vision (computer and human), this principle was apparently first realized in the mid-70's at the Massachusetts Institute of Technology (MIT) through pioneering research under the leadership of David Marr [Marr, 1982]. It was then that the foundations of modern computer vision were set with attention shifting from restrictions on the domain of application of a vision system to restrictions on visual abilities (autonomous parts). The focus of current research is defined more in terms of topics that correspond to identifiable modules in the human visual system: and although it is not clear what such modules in the human visual system are, research in neurobiology, neuroanatomy, psychophysics and psychology [Weiskrantz et al., 1974; Stevens, 1981; Marshall et al., 1973; Gibson, 1950; Land et al., 1971][3] has provided strong evidence that cues such as shading (image intensity variations), texture (distribution of surface markings), contours and line drawings, motion and stereo are very helpful in understanding properties of 3-D surfaces from their visual images.

The change of focus in research from a narrowly specified domain to a specific module (not necessarily present in the human visual system) had several consequences on the way research in understanding vision is conducted. One such consequence has been the decline in the construction of entire vision systems, i.e., systems that exhibit some vertical integration and use knowledge at all levels including domain specific information. "In order to complete the construction of such systems, it is almost inevitable that corners be cut and many overly simplified assumptions be made", as described in [Brady, 1982]. This will result in a system capable of carrying out a limited number of applications (which might be beneficial to industrial systems), that do not enhance our understanding of vision in general.

As we have suggested in the past [Aloimonos et al., 1988], there are basically two schools of thought in the way research in computer vision is performed today, as shown in Figure 1.

The reconstruction school worries about the reconstruction of the physical parameters of the visual world, such as the depth or orientation of surfaces, the boundaries of objects, the direction of light sources and the like. The recognition school worries about the recognition or description of objects that we see and involves processes whose end product is some piece of behavior like a decision or a motion. The recognition school is goal directed, and from this point of view it might be classified, in a sense, as a bottom-up Marr paradigm school, while the reconstruction school can be considered as top-down in the Marr paradigm.

As used here, a top-down methodology (top-down in the Marr paradigm) refers to an approach which starts with the development of a general theory for some visual process. The hope is that the theory will provide useful insights into a number of specific applications. This approach has the advantage that it is not tied to any particular application. The corresponding disadvantage is that, because the framework within which the theory is developed is always a simplification of the real world, theoretical development may not always lead to practically usable techniques.

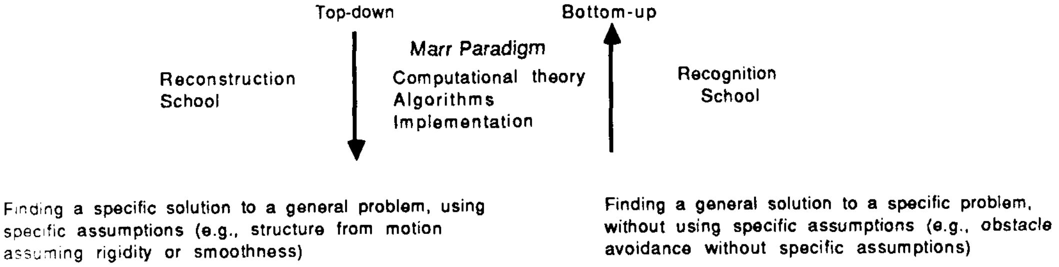

Top-down
Bottom-up
Marr Paradigm
Reconstruction School
Computational theory
Algorithms
Implementation
Recognition School
Finding a specific solution to a general problem, using specific assumptions (e.g., structure from motion assuming rigidity or smoothness)
Finding a general solution to a specific problem, without using specific assumptions (e.g., obstacle avoidance without specific assumptions)

Figure 1.

[3] For example, one rich source of evidence for the existence of modules in the human visual system is apparent from the study of patients with disabilities that come from brain lesions. On the other hand, psychophysicists perform experiments where a particular module of the human visual system is seemingly isolated, such as Julesz's experiment on stereoscopic fusion without monocular cues, Land's demonstration on the computation of lightness, Gibson's experiments on the perception of shape from texture, etc.

A bottom-up approach (bottom-up in the Marr paradigm), on the other hand, starts by developing systems which actually perform certain practical tasks. Here the hope is that commonalities observed among several such systems will allow the eventual formulation of more general principles. This approach has been criticized on the grounds that it will produce results of too narrow a scope, and without adequate theoretical foundation. On the other hand, if it produces anything at all, it is guaranteed to produce results which can actually be applied. Moreover, a solution to a specific problem can certainly have a solid theoretical foundation within the domain of its applicability. The success of such an approach depends on the appropriate selection of the specific problems. In particular, the problems must have a kind of environmental invariance which makes their solution applicable in a wide range of situations.

The situation can be viewed in terms of the well known paradigm described by David Marr [Marr, 1982]. The paradigm states that a machine performing an information processing task must be understood on three levels: the level of computational theory, the level of representation and algorithms, and the level of the hardware implementation. A top down approach thus corresponds to starting with the computational theory and developing software and hardware to match, whereas a bottom-up approach starts with working software and hardware and attempts to devise an appropriate theory.[4]

Research in the reconstruction school (top-down in the Marr paradigm) can be considered as research for finding a *specific* solution to a *general* problem. Working on visual modules, such as shape from shading (or shape from x in general), structure from motion and the like, someone trying to develop a computational theory for the perceptual process at hand has to make specific assumptions (for example, knowledge of the reflectance map in shape from shading, rigidity in structure from motion, knowledge of 3-D texel distribution in shape from texture and the like). On the other hand, if one works bottom-up in the Marr paradigm, i.e., wishes to develop a system for a specific task, then she should perceive her problem as one of finding a *general* solution to a *specific* problem. Let us consider an example in order to clarify matters, in particular the problem of obstacle avoidance (visually). If in the reconstruction school of thought (top-down in the Marr paradigm) the modules of stereo, structure from motion, etc., are highly developed, then the solution to the obstacle avoidance problem comes as a simple application. However, one might not need any of these modules in order to avoid obstacles. One might be able to solve this specific problem (obstacle avoidance) by developing a *general* solution that does not use any specific assumptions (for such an approach see Nelson and Aloimonos, 1988a).

Before we proceed with an outline of the paper, in order to avoid leaving the reader with a sense of dissatisfaction, we present some examples of recent work that fit in the previously introduced classification.

Top-down in the Marr paradigm includes the works of Horn on shape from shading and passive navigation; the works of Poggio and his colleagues on stereo, motion and zero crossings; the works of Witkin, Bajcsy, Stevens, Kanatani and Gibson on shape from texture; Brady's, Kanade's and Kender's work on shape from contour; the works of Binford and his colleagues on stereo, edge detection and specularities; the works of Shafer and Kanade on shadows and color; the regularization and segmentation works of Poggio, Mumford, Blake, Terzopoulos, Nagel, Pavlidis, Lee, Grimson, Marroquin, Geman and Geman; the works on computing optic flow; Huang's work on estimating 3-D motion; Ullman's and Faugeras' work on structure and motion estimation; Rosenfeld's work on relaxation, etc.

Bottom-up in the Marr paradigm includes the works of Brooks on achieving AI through building robots, the works of Grimson and Lozano-Pere̓z on robotics applications, the works of Nelson and Aloimonos on navigation, the work of Solina and Bajcsy on recognition, the work of Nevatia on recognizing runways from aerial images of airports, the work of Kanade and his colleagues on navigation in constrained domains, Ikeuchi and Horn's work on bin picking, and in general works concentrating on a specific application, such as problems involved in automated assembly, fault recognition of specific machine parts, etc.

At this point it is worth noting that due to the inability of the general theories developed in the Marr para-

[4] One might ask: how can you have software and hardware without some form of computational theory? What is actually meant here is that you have some very simple and intuitive computational theory, which is promising but very hard to analyze in a rigorous way. Experimenting and hoping that software based on this idea might be successful, you might end up with a working mechanism. Then, you will have to prove why it works and this will be the computational theory. An example here will help clarify the bottom-up approach. Consider the problem of visual homing, i.e. finding a specific position in an environment from any arbitrary position in this environment, on the basis of visual information. A simple idea would be to store several views of this environment (actually some pattern from the view, in order to reduce space requirements) and associate with each one a motion toward the home point (that has zero motion associated with it). Then while performing homing, the automaton would have to match (approximately) sensored images (patterns) with the best one in the memory and take the appropriate motion. Nelson (1988) experimented with this simple idea and it turned out that it was very successful. Then, Nelson created a computational theory that, given the environment, will determine how many views are necessary to store, and given statistics from the environment, what is the probability of successful homing, what is the optimal threshold for approximate matching, what are the optimal patterns, etc.

digm (top-down)[5] to find applications in practical systems (navigation and object recognition—two problems in vision that almost any published research paper addresses directly or indiiectly) some researchers[6] call for system development, even if we don't understand the individual components. Such an approach, if attempted, should be done bottom-up in the Marr paradigm.[7] The reason for the limited success of the theories developed in the Marr paradigm is that most modules are confronted with the task of solving ill-posed problems. The ill-posedness comes from either the lack of stability or from the lack of information necessary to guarantee a unique solution. In the sequel some of these problems are addressed. First, an example from our research is presented in the study of structure from motion (Section 2), and emphasis is placed on the analysis of robustness of proposed algorithms. Section 3 is devoted to the description of a theory of discontinuous regularization that can serve both for unifying and for integrating visual modules. The paper concludes with a list of future research problems.

2. STRUCTURE FROM MOTION: FACT OR FICTION?

The problem of structure from motion[8] (sfm) has been extensively studied in the recent literature. A plethora of algorithms for solving the problem has been suggested and with the exception of very few that solve part of the problem in specific domains and using specialized sensors, none of the techniques has found any realistic applications. Why? I do not intend to present here a treatise on sfm and analyze previous work. Instead, I will concentrate on the approaches whose first step is to establish retinal correspondence and whose second step is to use this correspondence in order to recover structure and 3-D motion, and describe parts of a complexity and stability analysis.

2.1. ESTABLISHING CORRESPONDENCE

This is a hard problem. Several techniques have been proposed that work only when the assumptions employed are true in the real world. The basic problem is that all approaches (implicitly or explicitly) assume that the motion field is smooth (with the exception of the ones working in specific domains). But motion fields in real scenes are full of discontinuities, and reconstruction of nondifferentiable (or discontinuous) functions is a mathematical question whose answer is not completely known. The theory of discontinuous regularization that we are developing can be used for establishing correspondence and I will postpone such an exposition until Section 3, in order to avoid duplication. Here, I would like to touch on the complexity of the problem, and show that it is intrinsically connected to segmentation.[9] Since the goal is a negative result, it will suffice to assume knowledge of the 3-D motion, i.e. to consider stereo correspondence, in a piecewise planar world.

We use the following notation: given two finite two-dimensional images I_l and I_r, and two finite sets of feature points in the two images, P_l and P_r, a *correspondence* is defined as discrete mapping $\delta_m : P_l \rightarrow P_r$. Since the two sets of features points are finite, each point can be considered an ordered pair (x, y) with integer or rational coordinates. Let $|P_l| = n$ and $|P_r| = m$. We also assume the images are registered with horizontal epipolar lines, so that the disparity δ of a match $\delta_m\big((x_l, y_l)\big) = (x_r, y_r)$ can be defined as $\delta(x_l, y_l) = x_r - x_l$ and $y_l = y_r$.

A *surface covering* is defined as a set covering $S = \{s_1, s_2, \ldots, s_k\}$ of the feature points P_l in the left image. A surface covering is interpreted as the segmentation of the feature points into surfaces. Given a correspondence δ_m, a *planar surface covering* over that matching is defined as a surface covering where every subset of points s_i has matches which can be fit exactly by a plane in three-space. This implies that for each s_i, the disparity data $\big(x_l, y_l, \delta(x_l, y_l)\big)$ can be fit exactly by a *linear disparity functional* $\delta_i(x_l, y_l) = a + bx_l + cy_l$.[10]

[5] Such as shape from x, structure from motion, optic flow, etc.

[6] See *Proceedings of Image Understanding Workshop* 1986, 1987, 1988.

[7] One might maintain that building systems even if we don't understand components contributes to our understanding of vision (from the experience gained from repeated failed attempts). That is true, but such a methodology is very expensive.

[8] Recovering the structure and 3-D motion of a moving object from a sequence of its images.

[9] Eastman, private communication.

[10] Note that some intuitively reasonable restrictions have not been placed on correspondences or surface coverings. Correspondences could be required to be 1-1, so each feature point has only one match; and surface coverings could be required to be eliminate transparent planes. These simplifications allow us to avoid issues like points without matches due to occlusion.

Consider now three problems of interest.

Plane matching problem. Given feature point sets P_l, P_r from images I_l, I_r, and a surface covering S_m, find a correspondence δ_m over which S_m is a planar surface covering; or, determine that there is no such correspondence.

Planar point covering problem. Given feature point sets P_l, P_r from images I_l, I_r, and a correspondence δ_m, find a planar surface covering S_m such that the cardinality K of the covering is minimized.

Plane matching covering problem. Given feature points sets P_l, P_r from images I_l, I_r, find a correspondence δ_m with a planar surface covering S_m over δ_m such that the cardinality K of the covering is minimized.

In the first problem, we know a covering and wish to find a correspondence; in the second, we know a correspondence and wish to find a covering; in the third, we know nothing in advance and wish to find both a covering and a correspondence. The first problem is polynomial in complexity,[11] while the second[12] and third are NP-hard. However, the second and third are polynomial *if we know the number of planar surfaces, K, in advance*. This implies that segmentation into an unknown number of surfaces, not matching, is the expensive part of correspondence.

The significance of results such as the above is that the "segmentation" part of the correspondence process is the expensive one. Clearly then, one has two choices in attacking the correspondence problem: to either develop retinal motion algorithms that at the same time attempt to couple discontinuities in the flow field with discontinuities in the world (i.e. work like that of Poggio [1988], Geman and Geman [1986], Mumford and Shah, Shulman and Aloimonos, Blake and Zisserman, Nagel and Enkelman), or devise good heuristics for solving the problem in specific domains.

2.2. COMPUTING STRUCTURE AND MOTION IN THE PRESENCE OF NOISE

Several algorithms dealing with structure and motion estimation from discrete frames have been published. The imaging geometry is the usual one: Coordinate system $OXYZ$; image plane $Z = 1$; O is the nodal point. A point P in 3-D is represented by its position vector $[X \;\; Y \;\; Z]^T$ and its image on the image plane by $P \cdot \frac{1}{Z} = \left[\frac{X}{Z} \;\; \frac{Y}{Z} \;\; 1\right]^T$ in 3-D coordinates and $\left[x \;\; y\right]^T$ in image plane coordinates (Figure 2).

[11] *Simple proof.*

[12] Planar point covering with unknown K is NP-hard. The proof follows by reduction of linear point covering to planar point covering. Linear point covering was defined and proved NP-hard in [Megiddo and Tamir, 1982]. Their definition follows, with rational or integer coordinates understood. "*(Linear) point covering problem.* A set of points $(x_1, y_1), \ldots, (x_p, y_p)$ is given. Find a collection of straight lines $\{l_1, \ldots, l_r\}$ of minimum cardinality, such that (x_i, y_i) lies on at least one l_j."

A direct reduction is possible by extending each line in the two-dimensional linear covering problem into a plane in the three-dimensional planar covering problem. Let P_2 be the set of points in an instance of linear point covering. Transform P_2 to the set P_3 by embedding each original point (x_i, y_i) in three dimensions as the point $(x_i, y_i, 0)$. Now construct the sets P_e^- and P_3^+ with points (x_i, y_i, z_i^-) and (x_i, y_i, z_i^+), respectively, such that $P = P_3^- \cup P_3^+$ meets the following condition: any four distinct points from P are coplanar only if the four original points from P_2 are collinear. A set of collinear points in P_2 will transform into a set of coplanar points in $P \cup P_3$ that can be covered by a vertical plane parallel to the z-axis. If K lines can cover the points in the original set P_2, then the same lines transformed into K vertical planes will cover the points in $P \cup P_3$. Conversely, $P \cup P_3$ cannot be covered in less than K planes; in fact, P cannot be covered in less than K planes. The possible planes fall into two groups—vertical and non-vertical. Less than K vertical planes cannot cover P, for then less than K lines would cover P_2, a contradiction. Less than K non-vertical planes cannot cover P, for every non-vertical plane covers only three points so $2n/3$ would be required to cover all points in P; but K must be less than $n/2$. Less than K vertical and non-vertical planes cannot cover P, for either the two classes of planes are disjoint, in which case the previous argument applies to the subset of non-vertical planes; or the two subsets are not disjoint, in which case any non-vertical plane which shares a point with a vertical plane can be replaced by a vertical plane that covers the one or two points not shared with the first vertical plane.

To finish the reduction, we need to show that the coordinates z^+ and z^-, satisfying the condition above, can be computed in polynomial time. The computation can be done by selecting the points from P_2 one at a time, and assigning first z^- and then z^+ to each point such that the condition is still satisfied. First, order the points in P_2 in any order as long as the first three points are not collinear. If there are not three non-collinear points from P_2, then P_2 can be covered by a single line and any assignment of non-zero coordinates to the z coordinates will do. If there are three such points, then assign non-zero values to the z coordinates of the three so the six new points satisfy the non-coplanar requirement. Now, take the remaining points from P_2 in order and assign first z^- and then z^+ to each point, comparing each new coordinate to all previously existing sets of three points in P to avoid creating a coplanar set. For each new coordinate, this creates up to $\binom{i}{3} = O(n^4)$ forbidden values, where i is the number of previously assigned coordinates; this brute force approach would take $\sum_{i=3}^{2N-1} \binom{i}{3} = O(n^4)$ time to generate all the forbidden values. However, a set of three points may be collinear in P_2 with the current point, and in this case the four new points will be unavoidably coplanar so the set places no constraint on the new coordinate values. Choosing three non-collinear points to start the process avoided the possibility of creating three or more points collinear in both P_2 and P which then forces a set of four or more coplanar points when any new, non-collinear point, is added to P.

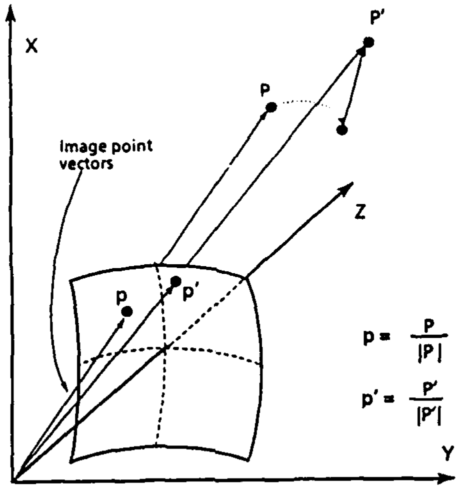


Figure 2. A point P in the 3-D rotates and translates to P' Its image then moves from p to p'. The image surface is a patch of a sphere so the image point vectors have length one

The vector $p = \dfrac{P}{||P||}$ contains as much information as $P \cdot \dfrac{1}{Z}$ and has the advantage of constant length. A point P in 3-D that projects on p translates by T and then rotates by R (R is a rotation matrix) to P' which projects to p'. The following relation then holds:

$$P' = R(P + T) \Rightarrow R^T \cdot P' = P + T \Rightarrow T \times (R^T \cdot P') = T \times P \Rightarrow P \cdot [T \times (R^T \cdot P')] = 0$$

or $(P, T, R^T \cdot P') = 0$ where $(\cdot, \cdot, \cdot)$ is the triple product. Dividing by $||P|| \cdot ||P'||$ we get $(p, T, R^T \cdot p') = 0$ or

$$p \cdot E \cdot p' = 0 \tag{1}$$

where $E = T_S \cdot R^T$,

$$T_S = \begin{bmatrix} 0 & -t_3 & t_2 \\ t_3 & 0 & -t_1 \\ -t2 & t_1 & 0 \end{bmatrix}$$

Eq. (1) is a linear equation with unknowns the elements of E. If we take at least 8 such equations we can almost always recover the motion parameters [Tsai and Huang, 1984]. To increase the stability we can take more than eight points and do least squares to minimize a quadratic of the form

$$x^T \cdot A \cdot x \rightarrow \min \tag{2}$$

where x is a nine-dimensional vector each element of which is an element of E (its columns one on top of the other form x) and A is a matrix that depends on the various pairs of points p_i, p'_i.

Least squares is the easiest method we can use. But it requires the variables (the elements of the vector x) to be independent. Here this is not the case; the solution that least squares finds, without taking into consideration the dependency, does not represent a matrix E that is decomposable into T_S and R^T. Even if from the solution that minimizes (2) we find a matrix E that is nearest to being decomposable, this might be far from minimizing (2) in the sense of finding R, T that do this. Another problem is the physical interpretation of what we minimize. Unless x is decomposable to R, T then there is no physical interpretation of the quantity we minimize.

So two things need to be done: First, use constraint minimization for (2), and second, find what the quantity we minimize stands for. Let us now introduce the error in correspondence in our calculations. A point P_1 moves to P_2 with rigid motion, $P_2 = R(P_1 + T)$. The correspondence algorithm matches it incorrectly with $P'_2 = P_2 + n$ or $P'_2 = R(P_1 + T) + n$ where n is the error vector. Proceeding as before we get finally

$$p_1 \cdot E \cdot p'_2 = (p_1, T, R^T \frac{n}{||P_2||}) \quad \text{or} \quad p_2 \cdot E \cdot p'_2 = (p_1, T, n') \tag{3}$$

where

$$n' = R^T \frac{n}{||P_2||} \tag{4}$$

The l.h.s. of (3) is what we minimize in (2). So this minimization process minimizes a function of the correspondence error. The r.h.s. of (3) equals

$$(p_1 \times T) \cdot n' \tag{5}$$

First notice that we minimize the component of the error that is parallel to $p_1 \times T$. The other component is irrelevant to the estimation of 3-D motion and affects only the estimation of the structure of each point; hence depth estimation for each of these points is at the mercy of the error in the pair of its projections (see Figure 3). Needless to say, trying to minimize both components of the error is impossible. Second, far-away points have less weight because in (4) $||P_2||$ is in the denominator. What we can actually minimize is one component of the image of the noise vector.[13]

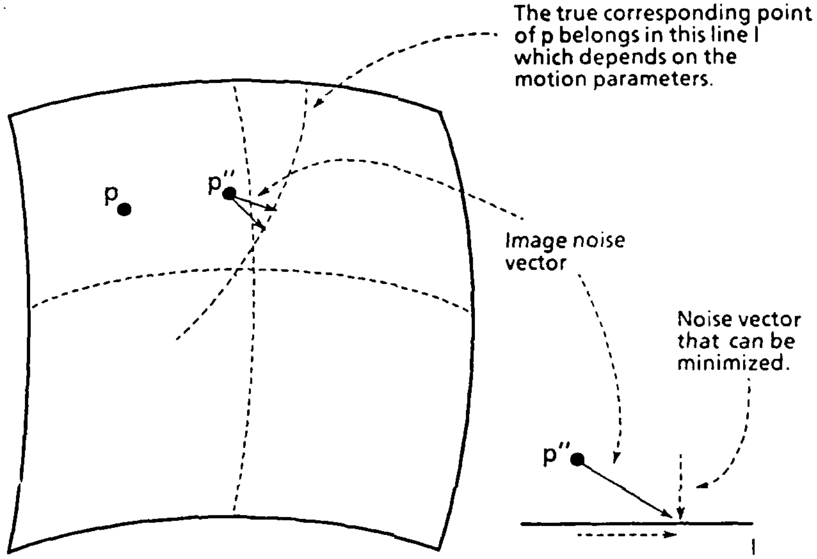

The true corresponding point of p belongs in this line l which depends on the motion parameters.
p
p''
Image noise vector
Noise vector that can be minimized.
p''
l

Figure 3. We have to find a set of motion parameters that minimize the distance of the points p'' from their corresponding lines l. This distance is the component of the noise we can minimize.

One of the difficulties inherent in estimating the motion is related to the size of the object observed. When the object is both small and almost planar then pure translation and pure rotation may create very similar flow patterns or correspondence pairs. This phenomenon appears here too. In (5) the error is multiplied by the sine of the angle between p_1 and T. When the field of view is small then the vectors p_i of the points form a tight bundle. Then a choice of T somewhere between them makes the sine of the angle between T and the points very small. Since this sine is multiplied by the error the result is a small number (Figure 4). If the noise is sufficiently large then the solution of T is biased towards being pointed to the object. Part of the blame here goes to the sine that appears in (5).

As a conclusion we can say the following.

- No matter how many points we use we cannot reduce the error in structure estimation using *two frames*. This problem cannot be cured with two frames.
- When the field of view is small and there is noise, the translation is biased towards the observed object. Ultimately, this means high error in the output, because our estimator is biased.

Having established the fact that computing structure using two frames in the presence of noise is completely at the mercy of the error in retinal correspondence in such a way that reducing the error is simply impossible in general, we describe our approach to answering the following question:

(a) How can we optimally estimate 3-D motion using two frames? (i.e., what is the best we can do, using two frames?)

[13] The above two things are natural and both of them are to be expected.

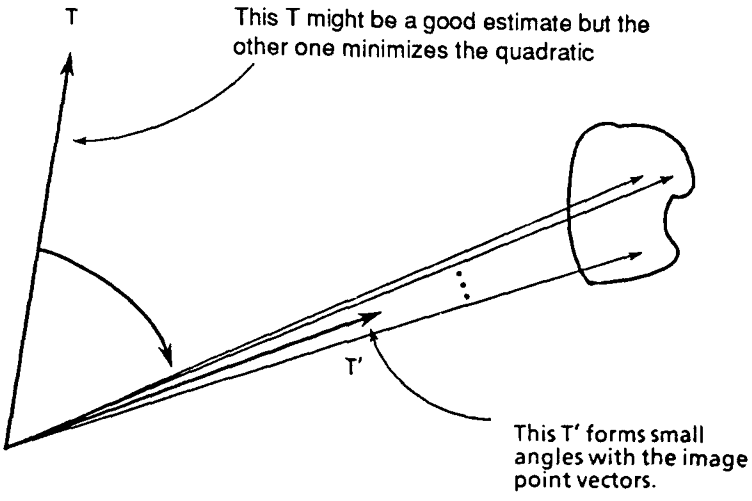

T
This T might be a good estimate but the other one minimizes the quadratic
T'
This T' forms small angles with the image point vectors.

Figure 4. An estimate of translation T that minimizes the quadratic just because it forms small angles with the image point vectors might be preferred over one that minimizes the overall image error if a non-optimal technique is used.

(b) How could we cure the problematic case of structure estimation from motion sequences?

The forthcoming analysis follows [Spetsakis and Aloimonos, 1988, 1989].

2.3. OPTIMAL MOTION ESTIMATION FROM TWO FRAMES

We present the existing approach to the problem of motion estimation with some comments on the difficulties associated with it, and then the solution (to those difficulties not inherent in the problem itself) which is optimal in the sense that it finds the absolute minimum of a *function* of the overall detectable error in the correspondence.

Definition: We define the *vector* of a 3×3 matrix E to be $V(E)$, a vector of dimension 9 whose elements are the same as the elements of the matrix E and ordered so that they are the columns of E one on top of the other..

Tsai and Huang [1984] developed an algorithm that finds T and R, given a matrix E for which there exist a vector T and a matrix R such that

$$E = T_S \cdot R^T$$

where

$$T_S = \begin{bmatrix} 0 & t_3 & -t_2 \\ -t_3 & 0 & t_1 \\ t_2 & -t_1 & 0 \end{bmatrix}, \quad T = \begin{bmatrix} t_1 \\ t_2 \\ t_3 \end{bmatrix}$$

They proved that there are two solutions and the algorithm can find both. Furthermore the algorithm is very stable in presence of noise, partly making up for the extreme instability of the process of finding the matrix E (overall, though, the algorithm behaved poorly due to the difficulties in finding E [Tsai and Huang, 1984]).

Below we rephrase the algorithm in our notation and we prove that the R, T their algorithm finds are such that $|| \, V(E) - V(M) \, || = \min$ where $M = T_S \cdot R^T$.

Algorithm: Let E be a 3×3 matrix. We find T, R as follows:
If the *SVD* (singular value decomposition) of E is

$$E = U \begin{bmatrix} \sigma_1 & & \\ & \sigma_2 & \\ & & \sigma_3 \end{bmatrix} V^T \ , \ 0 \le \sigma_1 \le \sigma_2 \le \sigma_3$$

then T is parallel to the first column U_1 of U and $\| T \| = \frac{\sigma_2 + \sigma_3}{2}$. Matrix T_S has two degei.erate singular vectors and one parallel to U_1. So one of its possible SVD's is $T_S = U \begin{bmatrix} 0 & & \\ & \| T \| & \\ & & \| T \| \end{bmatrix} V^T$ Then R is $R^T = V^T \begin{bmatrix} s & & \\ & s_1 & \\ & & s_1 \end{bmatrix} V_t$ where $s = \det(V_T) \cdot \det(V)$ and $s_1 = \pm 1$.

Theorem: *The R, T computed above satisfy* $\| \mathrm{V}(E) - \mathrm{V}(M) \| = \min$ *where* $M = T_S \cdot R^T$.

Proof: (see [Spetsakis and Aloimonos, 1988]).

2.3.1. Solving the Constraint Minimization Problem

The mathematical problem at hand is to find a nine-dimensional vector x such that $x^T \cdot A \cdot x \rightarrow \min$ and the matrix whose vector x is to be decomposable to R, T as described above. The constraint is clearly non-linear and very difficult to be written down analytically. We describe here two methods to treat the problem: one is a variation of Newton's method and the other is a decomposition of the problem into two parts, thus reducing the dimensionality. Both of them are efficient.

2.3.1.1. First method. We present the method along with a sketch of a proof of convergence. Let $x = \mathrm{V}(E)$

$$E = T_S \cdot R^T = \begin{bmatrix} 0 & t_3 & -t_2 \\ -t_3 & 0 & t_1 \\ t_2 & -t_1 & 0 \end{bmatrix} \cdot \begin{bmatrix} r_1 & r_4 & r_7 \\ r_2 & r_5 & r_8 \\ r_3 & r_6 & r_9 \end{bmatrix} = \begin{bmatrix} t_3 r_2 - t_2 r_3 & t_3 r_5 - t_2 r_6 & t_3 r_8 - t_2 r_9 \\ t_1 r_3 - t_3 r_1 & t_1 r_8 - t_3 r_7 & t_1 r_9 - t_3 r_7 \\ t_2 r_1 - t_1 r_2 & t_2 r_7 - t_1 r_5 & t_2 r_7 - t_1 r_8 \end{bmatrix} .$$

Therefore

$$x = \begin{bmatrix} t_3r_2 - t_2r_3 \\ t_1r_3 - t_3r_1 \\ t_2r_1 - t_1r_2 \\ t_3r_5 - t_2r_6 \\ t_1r_8 - t_3r_7 \\ t_2r_7 - t_1r_5 \\ t_3r_8 - t_2r_9 \\ t_1r_9 - t_3r_7 \\ t_2r_7 - t_1r_8 \end{bmatrix} = \begin{bmatrix} r_1 & 0 & 0 & r_2 & 0 & 0 & r_3 & 0 & 0 \\ 0 & r_1 & 0 & 0 & r_2 & 0 & 0 & r_3 & 0 \\ 0 & 0 & r_1 & 0 & 0 & r_2 & 0 & 0 & r_3 \\ r_4 & 0 & 0 & r_5 & 0 & 0 & r_6 & 0 & 0 \\ 0 & r_4 & 0 & 0 & r_5 & 0 & 0 & r_6 & 0 \\ 0 & 0 & r_4 & 0 & 0 & r_5 & 0 & 0 & r_6 \\ r_7 & 0 & 0 & r_8 & 0 & 0 & r_9 & 0 & 0 \\ 0 & r_7 & 0 & 0 & r_8 & 0 & 0 & r_9 & 0 \\ 0 & 0 & r_7 & 0 & 0 & r_8 & 0 & 0 & r_9 \end{bmatrix} \cdot \begin{bmatrix} 0 \\ -t_3 \\ t_2 \\ t_3 \\ 0 \\ -t_1 \\ -t_2 \\ t_1 \\ 0 \end{bmatrix}$$

or $x = R_b \cdot T_b$, where R_b and T_b are the above matrix and vector respectively. T_b depends on the three translation parameters. R_b depends on the rotation matrix R which in turn depends on the three Rodrigues parameters [Bottema and Roth, 1979] b_1, b_2, b_3 of the rotatio.. So x is a function of a six-dimensional vector $\varsigma = [b_1, b_2, b_3, t_1, t_2, t_3]^T$. The Taylor series expansion of x is

$$x\ (\varsigma + \Delta\varsigma) = x\ (\varsigma) + A\ (\varsigma) \cdot \Delta\varsigma + 0(\Delta b_1^2) + 0(\Delta b_2^2) + \cdots + 0(\Delta t_3^2)$$

Matrix $A(\varsigma)$ is easy to construct. It has as columns the derivatives of x with respect to the elements of ς. The derivatives with respect to t_1, t_2, t_3 are obvious. The derivatives with respect to b_1, b_2, b_3 are $x_i = \frac{1}{2} (R_b + I) B_i (R_b + I) \cdot T_b$, $i = 1,2,3$ where the B_i's are

$$B_1 = \begin{bmatrix} 0 & & & & & & & & \\ & 0 & & & & & & & \\ & & 0 & & & & & & \\ & & & 0 & & & +1 & & \\ & & & & 0 & & & +1 & \\ & & & & & 0 & & & +1 \\ & & & -1 & & & 0 & & \\ & & & & -1 & & & 0 & \\ & & & & & -1 & & & 0 \end{bmatrix}$$

Similarly for B_2, B_3. The way to achieve convergence is to move in the column space of $A(\varsigma)$ so that the quadratic is decreasing in value. This, in general, will lead to values of x that do not satisfy the non-linear condition. But if l is the distance we moved in the column space of $A(\varsigma)$ then the distance of the nearest vector x that satisfies the non-linear condition is of order $O(l^2)$.[14] If we are not at a local extremum, the quadratic decreases by $O(l)$ and then increases by $O(l^2)$ and for sufficiently small l decreases overall. It is easy to prove that unless this process goes to a local extremum it eventually converges to a minimum.

2.3.1.2. Second method

This is a method that involves gradient descent in a 3-dimensional space. If there is a good guess for the solution, and in this case there is, then we need to move around only locally.

We have

$$T_b = \begin{bmatrix} 0 \\ -t_3 \\ t_2 \\ t_3 \\ 0 \\ -t_1 \\ -t_2 \\ t_1 \\ 0 \end{bmatrix} = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & -1 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \\ 0 & 0 & 0 \\ -1 & 0 & 0 \\ 0 & -1 & 0 \\ 1 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix} \cdot T = K \cdot T$$

Then the quadratic takes the form

$$x^T \cdot A \cdot x = T^T \cdot K^T \cdot R_b^T \cdot A \cdot R_b \cdot K \cdot T = T^T A' T \tag{6}$$

A' is a 3 X 3 matrix that depends only on the Rodrigues parameters of the rotation.

The value of T that minimizes the quadratic is a vector parallel to the eigenvector of A' that has the smallest eigenvalue. The value the quadratic assumes then is the smallest eigenvalue of A'. (There is a factor of two missing here: when $\| x \| = 1$ then the corresponding $\| T \| = \frac{\sqrt{2}}{2}$. When we minimize $x^T \cdot A \cdot x$ we silently assume $\| x \| = 1$ and when we minimize $T^T A' T$, $\| T \| = 1$. This causes no problem, though.)

Now the problem is really broken into two: One, computing the rotation parameters that minimize the smallest eigenvalue of the matrix A', and the other, minimizing a quadratic. The second is just an application of Rayleigh's principle, so there is an easy solution. For the first it is easy to use a modified Newton's method, because we can derive the analytic expression for the derivative. Recall that the minimal value of (6) is the smallest eigenvalue of A'. The derivative of A' with respect to the Rodrigues parameters of R_b is

$$\frac{dA'}{db_i} = \frac{1}{2} \cdot K^T \cdot \left[R_b^T \cdot A \cdot (R_b + I) \cdot B_i (R_b + I) + (R_b + I)^T \cdot B_i^T (R_b + I)^T \cdot A \cdot R_b \right] \cdot K$$

This is an unnecessarily complicated expression and can be simplified as follows: Form the matrix $A_r = R_b^T \cdot A \cdot R_b$ and fix the value of R_b at the current guess. To do gradient descent we can perturb A_r by pre- and post-multiplying by the matrix $R_p(b'_1, b'_2, b'_3)$ which is a function of b'_i's that now serve as unknowns. The initial guess for R_p is the identity matrix (e.g. all the Rodrigues parameters b'_1, b'_2, b'_3, are zero). The expression for the derivative is simplified because we take the derivative at the zero point of the parameters. So A' is now a function of three new parameters that we can perturb around zero. Thus

[14] This is why we needed to prove that the Tsai-Huang algorithm finds the nearest vector.

$$\frac{dA'}{db'_i} = 2\, K^T \cdot \left[B_i^T \cdot A_r + A_r \cdot B_i \right] \cdot K$$

The derivative of the smallest eigenvalue with respect to the i^{th} parameter is

$$\lambda^{(i)} = \phi^T \cdot \frac{dA'}{db'_i} \cdot \phi = 4\, \phi^T \cdot K^T \cdot B_i^T \cdot A_r \cdot K \cdot \phi$$

where ϕ is the eigenvector of the smallest eigenvalue. Using the modified Newton's method we can find the minimum. This method results in an algorithm that has a quite large basin of attraction so it works well if the initial guess is not that good. The main advantage though is that it can be modified to work with many frames.

2.3.2. Optimality

This is the most important question about the motion problem. First, because we want to do things as efficiently and robustly as possible; second, because we want to know how bad the best is! If the best is unstable, this is bad news, and a new direction of research should be looked for. If the best is stable, it is no news until an efficient way to compute it is developed. Here we are interested in optimal estimation techniques that lead to results that can be studied analytically. The maximum likelihood estimator is the best from this point of view. Assuming a Gaussian distribution it leads to a least squares formulation on which there is a lot of published work.

The maximum likelihood estimator is formulated as follows: Let $f\,(p_1, p_2; R, T)$ be the probability density that p_1, p_2 are a correspondence pair when R, T are the motion parameters. Then

$$\prod_i f\,(p_{i1}, p_{i2}; R, T)$$

is the probability that $\{\, (p_{i1}, p_{i2}) \mid i=1\ldots \,\}$ are the correspondence pairs. So if we find R, T that maximize this probability we have found the most typical solution.

Let now p_1 be an image unit vector in the first frame and p_2 in the second frame. If p_1, p_2 is a correspondence pair with R, T as motion parameters then $R^T p_2$ should lie on the plane defined by T and p_1. The error vector on the image has two orthogonal components which are assumed identically distributed. If p_2 is corrupted by an error n' its distance from the plane of T, p_1 is

$$\frac{\left[(T \times p_1) \cdot p_2 \right]}{\| T \times p_1 \|} = \frac{(T, p_1, n')}{\| T \times p_1 \|} = \frac{(T \times p_1) \cdot n'}{\| T \times p_1 \|} = \frac{\epsilon}{\| T \times p_1 \|}$$

As we see, only the component parallel to the unit vector $\dfrac{T \times p_1}{\| T \times p_1 \|}$ affects the distance from the T, p_1 plane. (The direction of this unit vector does not make any difference to the probability distribution because n' is uniformly distributed on the image.) So

$$f\,(p_1, p_2; R, T) = \alpha\, e^{-\frac{\left[\frac{\epsilon}{\| T \times p_1 \|} \right]^2}{2\sigma^2}}$$

where α and σ are constants and depend only on the noise distribution.

By using the standard procedure for maximum likelihood we find that we have to minimize the quantity

$$\sum_i \left[\frac{\epsilon_i}{\| T \times p_{1i} \|} \right]^2 \tag{7}$$

where i are the different points on the image. In the previous paragraphs we discussed the minimization of $\sum_i \epsilon_i^2$ (8) or to be more precise, we explicitly incorporated the restriction that $\| T \| = 1$

$$\sum_i \frac{\epsilon_i^2}{\| T \|^2} = \sum_i \frac{\epsilon_i^2}{T^T \cdot T}$$

and now we see that the optimum has some "weight" factor of $\dfrac{1}{\| T \times p_{1i} \|^2}$ in the minimization function.

This has two consequences: First, there is some weight in the equations different from 1. This has some small effect on the result. The second consequence is more important. Imagine the following situation: A small object on the z axis translates parallel to the x axis without rotation. Then the translation vector T that minimizes $\sum \epsilon_i^2$ in the presence of noise is parallel to the z axis because most of the p_1's of the object points are very close to the translation vector T so (p_1, p_2, T) is very close to zero no matter what the error is.

This way the solution tends to be parallel to the center of gravity of the p_{i1}'s when the noise level is rather high. This happens because we pick the eigenvector with the smallest eigenvalue which minimizes (8) but not (7). To incorporate the factor $\frac{1}{\| T \times p_{1i} \|^2}$ in the computation without increasing the complexity enormously we assume that the factor is uniform on the object and try to minimize the function

$$\frac{T^T A' T}{\| T \times C \|^2}$$

where C is the center of gravity of the points that constitute the object. This takes the form

$$\frac{T^T \cdot A' \cdot T}{T^T \cdot C' \cdot T} \tag{9}$$

where $C' = I - C \cdot C^T$, $C \cdot C^T$ is the outer product of C with itself and C is a unit vector. The Rayleigh principle would apply here too if C' were invertible, which it is not. To overcome this we use an approximate technique described in [Tsai and Huang, 1984].[15]

2.3.3. Relation to Other Approaches

One proposed approach for motion estimation [Prazdny, 1981] by Prazdny was based on the following observation: Since we know that the flow pattern of a pure translation is a set of lines converging to a point we can test different rotation matrices to derotate with until we find a flow pattern that looks like a pure translation. Prazdny, though, used an overly simplistic measure of similarity to the pure flow pattern. Here we choose the sum of the squares of the distances of the unit vector T from the planes defined by p_{1i}, p'_{2i}. (p'_{2i} is p_{2i} derotated and corrupted by noise.)

In order to find this distance k we find k such that $T + k \frac{p_1 \times p_2}{\| p_1 \times p'_2 \|}$ is coplanar with p_1, p'_2. We have

$$0 = \left| p_1, p'_2, T + k \frac{p_1 \times p'_2}{\| p_1 \times p'_2 \|} \right| = (p_1, p'_2, T) + \left| p_1, p'_2, k \frac{p_1 \times p'_2}{\| p_1 \times p'_2 \|} \right| = \epsilon + \frac{k}{\| p_1 \times p'_2 \|} (p_1, p'_2, p_1 \times p_2) = \epsilon + k \cdot \| p_1 \times p'_2 \|$$

So $k = \frac{-\epsilon}{\| p_1 \times p'_2 \|}$.

Although minimization of k as defined above is intuitively a good idea, it is better once again to use a maximum likelihood argument. The variance of k is approximately

$$\sigma_k^2 = \frac{\| T \times p_1 \|^2}{\| p_1 \times p'_2 \|^2} \bar{n}$$

where $\bar{n}$ is proportional to the variance of the error in the image. Proceeding as before we find that the quantity we want to minimize is

$$\sum \frac{\epsilon_i^2}{\| T \times p_{1i} \|^2}$$

Not surprisingly, it is the same as before.

We experimented with the above techniques using real images. For our experiments with real images we used an American Robot Merlin robot arm (having six rotational joints) in order to control the 3-D motion with high accuracy (in this experiment the camera is moving in a stationary environment). A Sony DC-37 CCD miniature television camera with a 16mm focal length lens was attached to the arm. The robot arm imaged the scene shown in Figure 5 (frame before the motion). Then, we moved the robot arm three inches in the horizontal direction only (translation along the other two axes was zero and rotation zero around all axes). The image after the motion is shown in Figure 6. We computed the correspondence of 14 points manually, using a cursor in the display, and these 14 vectors were the input to our algorithm for computing the three dimensional motion. It has to be noted, however, that although we computed the correspondence manually, there was error in the displacement vectors due to discretization. The algorithm recovered the direction of translation with an error of 3.6%.

[15] This approximation gives very accurate results in the case of a small object (small viewing angle). In the case of a larger viewing angle one has to use one of the standard routines that minimize non-linear functions. The problem with these is that one has to deal with each point in each iteration, which is expensive compared to the methods discussed above that just construct a 9 × 9 matrix and iterate on that.

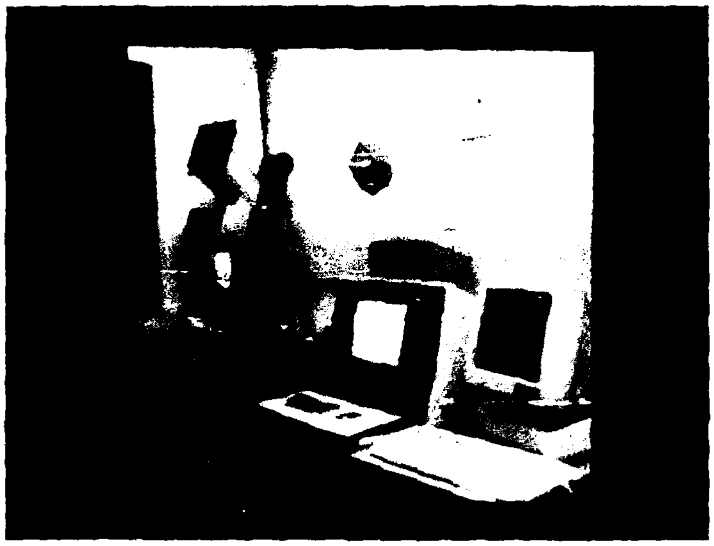


Figure 5.

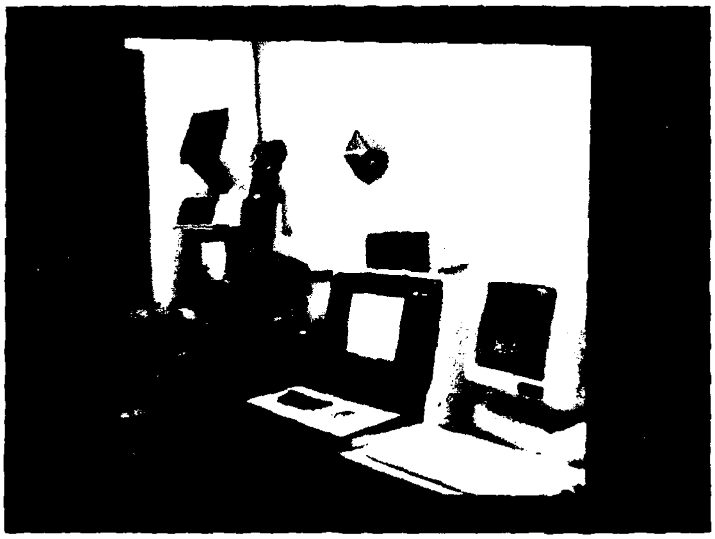


Figure 6.

The rotation was recovered very accurately. The reader can compare these results with results of similar experiments by Tsai and Huang where the correspondence was again computed manually and the translation was incorrectly recovered.[16]

[16] Consequences for camera calibrations are obvious.

2.4. A MULTI-FRAME APPROACH

The previous sections make the limitations of the two-frame approaches obvious. The sensitivity of such approaches seems to be inherent in the problem itself; in order to overcome it one has to use redundant information (more feature correspondences and more frames) and use it optimally. Work on more frames has been done by Shariat [1986], Matthies et al. [1988] and Young and Chellappa [1988] but they used restrictive assumptions. We present here an algorithm that does not employ any assumptions beyond rigidity. The analysis follows [Spetsakis and Aloimonos, 1988b].

From a first look it seems that the main advantage of multi-frame approaches is the increase of robustness of structure estimation, because, for each point such that we want to estimate its 3-D location, we have more information, while it has a minor effect on the estimation of interframe motion. This is not true, though. It has a very positive effect on both the structure and motion estimation. The reason for the latter is, simply stated, that one can get more constraints on the motion parameters using many frames than using a two-frame approach for every pair of frames.

We use the pinhole camera model and with the term image point vector we mean the position vector of the image point.

We assume the camera moves in a stationary environment and the motion between frames i and $i+1$ is described by a rotation matrix $R_{i,i+1}^T$ and a translation vector $-T_{i,i+1}$. These are the motion parameters we want to recover (and subsequently the object structure using them). We can equivalently assume that a rigid object moves in front of the camera with motion parameters $R_{i,i+1}$ and $T_{i,i+1}$. The input to the algorithm is the set of image points $p_{ij} = [p_{ijx}, p_{ijy}, p_{ijz}]^T$ (preferably of unit length) where i refers to the frame number and j to a point number in the i^{th} frame. Also the correspondence of the points is given, e.g. for every point in each frame all its corresponding ones in the other frames are given (in the ones where corresponding points exist). The output of the algorithm is the set of the motion parameters and the 3-D locations of the points, or to be more accurate ***the set of motion parameters and the set of 3-D points that projected onto successive frames moving according to these motion parameters, give images that are closest to the input images.***

Previous approaches employed assumptions that were either unrealistic or very restrictive for most applications. These included smooth, or even steady motion between successive frames, partially known structure or something equivalent like stereo, fixed number of frames or points etc. The approach here has many advantages that one would like to see in a motion algorithm. Among them are minimum assumptions, no requirement of a priori knowledge other than rigidity and flexibility (no restriction on the number of frames and the number of points in the frames).

2.4.1. Loaded Spring Model

The input data consist of N sets of image points that correspond to the N frames; each set contains a number of these points (the number of points may be different from set to set). An image point p_{ij} in the i^{th} frame or set is defined with respect to the coordinate system of the i^{th} frame, so within the set, the points have known relative position. By p_{ij} we can denote either the vector itself or the line that forms the extension of the vector, in other words the line of sight. One way to visualize the set of image point vectors with known relative positions is to imagine that the image vectors of the points form a rigid bundle of vectors passing through the same point, the origin. Each frame has a bundle, so there are N bundles whose relative positions are the unknown motion parameters.[17] Suppose now that we have a candidate set of motion parameters and we place the bundles in the corresponding relative positions. Suppose further that p_{ij} and $p_{i'j'}$ correspond. Then the extensions of the image point vectors of p_{ij} and $p_{i'j'}$ must meet each other at the actual 3-D point whose projections are p_{ij} and $p_{i'j'}$ (Figure 7). If they don't meet then there is some error involved either in the pinpointing of the corresponding points or in the guess of the motion parameters, and most probably the error involves a combination of both. A good measure of the error is the length of the common normal of these two lines, and of course a good estimate of the position of that point is somewhere in the vicinity of the common normal. In case the point has corresponding points in other frames as well then all the extensions of the position vectors of the image points must meet at the 3-D point that created the images. If the lines do not pass through the same point then the $\frac{n\cdot(n-1)}{2}$ common normals formed, under some mild condition, are of overall length greater than zero. This overall length represents the error introduced by the candidate solution. A good way to combine error functions in one minimization process is the mean square, so it is desirable to use the sum of the squares of the lengths of the common normals as the measure of their overall length. This quantity, as described above, measures the error associated with one

[17] The translation vector is the vector that spans the origins of the two frames and the rotation matrix is the matrix by which each vector in the one frame must be multiplied so that it becomes parallel to the same vector in the other system.

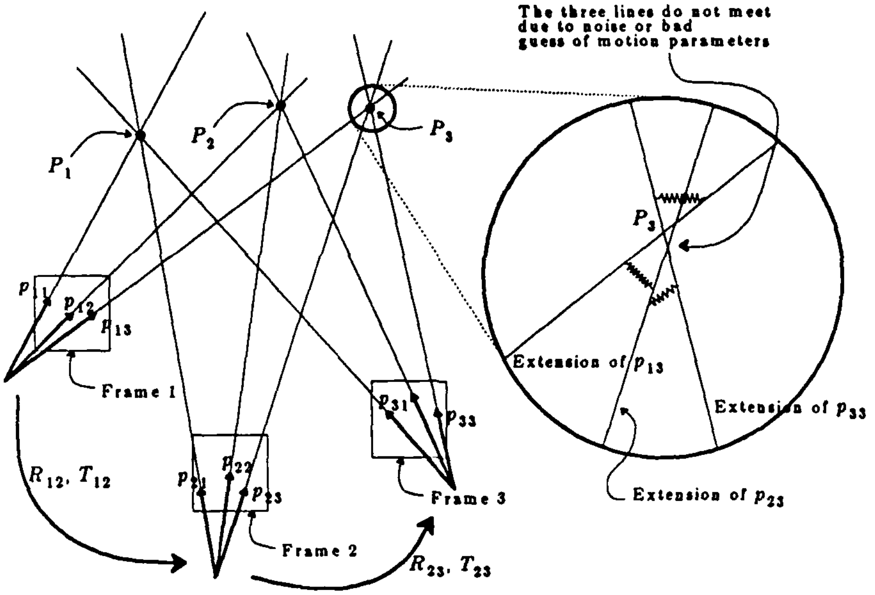


Figure 7. The springs tend to pull the frames in a position that minimizes their energy. In a noise free case they force all the lines to meet at their corresponding 3-D point.

point only. To include the rest of the points we just sum over all frames and all points. The error associated with the candidate solution is then a double summation of squared quantities. This summation has several very nice properties one of which is its resemblance to well studied expressions in other fields of science.

It is very typical in the physical sciences to have expressions that give the energy of a system as a function of its state that are quadratic with respect to the quantity that expresses this state; e.g. kinetic energy is proportional to the square of the velocity, elastic energy is proportional to the square of the increase of the length of the spring, electric energy is proportional to the square of the voltage across the capacitor, etc. Of course, if the system is more complicated, these expressions become sums of squares. Our choice to use sums of squares has the advantage of being related to the vast literature of physics, and indeed, as we can show, the minimization takes the form of a minimization of an eigenvalue of a matrix that depends on the rotation parameters. In addition, this gives us an idea of how to visualize the minimization process by considering a physical system that has the same mathematical model and as a consequence the same behavior. The physical model that best resembles this minimization process and is easy to visualize is the spring model.

The image point vectors in the same frame have known relative positions, so one can imagine them as a solid bundle of lines meeting at the origin of the frame. Along the length of each line in this solid bundle are freely sliding springs that tie the lines to other lines that correspond to the same point. The natural length of the springs is zero, which means that a spring left without external force would have zero length. The bundles are not tied together directly but only through the springs to the lines of the image point vectors, are free to move relative to each other, and initially are positioned according to the first guess of the motion parameters. The springs pull together lines that were supposed to pass through the same point, thus exerting a force on the bundles which will eventually come to rest at the state of minimum energy of the springs. The relative positions where the frames come to rest are the unknown motion parameters.

2.4.2. Description of the Computational Model

In this section we elaborate on the computational aspects of minimization and show how to reduce it to an eigenvalue minimization problem. As will become evident, this reduction decreases the dimensionality of the problem to half and at the same time simplifies it significantly. Before we present the derivations, there are two important things we should address: The first one is what the weights should be in taking the mean square error. The natural answer is that we want the points nearby to be heavily weighted and the far-away ones to be less heavily weighted. This is easy to handle. The other is that the measure of the error as described is proportional to the square of the scale factor, and unless we make it independent we will have trivial solutions of zero scale factor. We again have an easy solution, this time just by dividing by a quantity that is proportional to the square of the scale factor; the most handy choice is, of course, a quadratic in the translation vectors.

Let P_{ij} and $P_{i'j'}$ be two corresponding points in frames i, i' and let the motion between these two frames be $R_{i,i'}$. $T_{i,i'}$ (rotation and translation, respectively). Then the length of the common normal of the lines that are the extensions of the image point vectors P_{ij} and $P_{i'j'}$ is given by

$$\frac{\left((R_{i,i'} \cdot P_{ij}) \times P_{i'j'}\right) \cdot T_{i'j'}}{\| (R_{i,i'} \cdot P_{ij}) \times P_{i'j'} \|} \tag{10}$$

The quantity in the denominator of (10) takes small values when the 3-D point is far away from the camera in those two positions and larger values when the point is closer. If we multiply the length of the common normal by this sine then we get the effect we wanted on the weighting of the points: far-away points have less weight than close-by ones. This also simplifies expression (10), since the denominator is cancelled. (This is not the optimum as described in [Spetsakis and Aloimonos, 1988] but is very close to it and much simpler.) After eliminating the denominator, (10) becomes $\left((R_{i,i'} \cdot P_{ij}) \times P_{i'j'}\right) \cdot T_{i,i'}$ or $(T_{i,i'}, R_{i,i'} \cdot P_{ij}, P_{i'j'})$, which leads to the well known formula in [Tsai and Huang, 1984]

$$p_{ij}^T \cdot E_{i,i'} \cdot p_{i'j'} \tag{11}$$

where

$$E_{i,i'} = R_{i,i'}^T \cdot \begin{bmatrix} 0 & t_3 & -t_2 \\ -t_3 & 0 & t_1 \\ t_2 & -t_1 & 0 \end{bmatrix}_{i,i'} = \begin{bmatrix} r_7t_2 - r_4t_3 & r_1t_3 - r_7t_1 & r_4t_1 - r_1t_2 \\ r_8t_2 - r_5t_3 & r_2t_3 - r_8t_1 & r_5t_1 - r_2t_2 \\ r_9t_2 - r_6t_3 & r_3t_3 - r_9t_1 & r_6t_1 - r_3t_2 \end{bmatrix}_{i,i'}$$

and t_1, t_2, t_3 are the x, y and z axis components of the translation vector $T_{i,i'}$; r_1, $r_2 \cdots r_9$ are similarly the components of the $R_{i,i'}$ rotation matrix

$$R_{i,i'} = \begin{bmatrix} r_1 & r_2 & r_3 \\ r_4 & r_5 & r_6 \\ r_7 & r_8 & r_9 \end{bmatrix}_{i,i'}$$

Expression (11) gives the error introduced by two corresponding points in the different frames due to either a bad guess or an error in correspondence. The mean square error can be given in a nice form if we write the square of (11) as

$$(p_{ij}^T \cdot E_{i,i'} \cdot p_{i'j'})^2 = e_{i,i'}^T \cdot a_{i,ji'} \cdot a_{i,ji'}^T \cdot e_{i,i'} \tag{11a}$$

where the e's and a's are nine-dimensional vectors. Vector $e_{i,i'}$ is constructed by stacking columns of the matrix $E_{i,i'}$ one on top of the other and $a_{i,ji'}$ is constructed by stacking the elements of the matrix $p_{i'j'} \cdot p_{ij}^T$ one on top of the other. Then the mean square error associated with all the points in two frames is

$$e_{i,i'}^T \cdot A_{i,i'} \cdot e_{i,i'} \tag{12}$$

where $A_{i,i'} = \sum_j a_{i,ji'} \cdot a_{ijí'}^T$ is a 9×9 matrix with its elements being known quantities. Vector $e_{i,i'}$ is a 9×1 vector whose elements are the functions of the unknown quantities (the motion parameters) which substituted in (12) should minimize it.

The obvious solution to this problem, namely the eigenvector of $A_{i,i'}$ that corresponds to the smallest eigenvalue, is not the answer. The reason is that the elements of vector $e_{i,i'}$ are not independent because they constitute nine non-linear functions of six motion parameters. Thus this is a non-linear constraint minimization problem and when noise is present we cannot neglect the constraint, the practical reason for this being that the solution to the vector $e_{i,i'}$ we find corresponds to a matrix $E_{i,i'}$ that cannot be decomposed into a rotation matrix $R_{i,i'}$ and a translation vector $T_{i,i'}$, and if we find the closest decomposition to a set of motion parameters it cannot be guaranteed that this is a minimum for expression (12).

Since our goal is non-linear constraint minimization, let's treat it as such and carry out all the possible simplifications. The expression for which the minimum has to be found is

$$e_{i,k}^T \cdot A_{i,k} \cdot e_{i,k} \tag{13}$$

which is a measure of the error involved in our correspondence process for frames i and k. As mentioned above, vector $e_{i,k}$ is not just any vector but has a special form and can be written as

$$e_{i,k} = \begin{bmatrix} r_7 t_2 - r_4 t_3 \\ r_8 t_2 - r_5 t_3 \\ r_9 t_2 - r_6 t_3 \\ r_1 t_3 - r_7 t_1 \\ r_2 t_3 - r_8 t_1 \\ r_3 t_3 - r_9 t_1 \\ r_4 t_1 - r_1 t_2 \\ r_5 t_1 - r_2 t_2 \\ r_6 t_1 - r_3 t_2 \end{bmatrix}_{i,k} = \begin{bmatrix} r_1 & r_4 & r_7 & & & & & & \\ r_2 & r_5 & r_8 & & & & & & \\ r_3 & r_5 & r_9 & & & & & & \\ & & & r_1 & r_4 & r_7 & & & \\ & & & r_2 & r_5 & r_8 & & & \\ & & & r_3 & r_6 & r_9 & & & \\ & & & & & & r_1 & r_4 & r_7 \\ & & & & & & r_2 & r_5 & r_8 \\ & & & & & & r_3 & r_6 & r_9 \end{bmatrix}_{i,k} \cdot \begin{bmatrix} 0 \\ -t_3 \\ t_2 \\ t_3 \\ 0 \\ -t_1 \\ -t_2 \\ t_1 \\ 0 \end{bmatrix}_{i,k} = R_{bik} \cdot \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & -1 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \\ 0 & 0 & 0 \\ -1 & 0 & 0 \\ 0 & -1 & 0 \\ 1 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix} \cdot T_{i,k} = R_{bik} \cdot K \cdot T_{i,k}$$

Now we can write expression (13) as

$$e_{i,k}^T \cdot A_{i,k} \cdot e_{i,k} = T_{i,k}^T \cdot K^T \cdot R_{b\,i,k}^T \cdot A_{i,k} \cdot R_{b\,i,k} \cdot K \cdot T_{i,k} = T_{i,k}^T \cdot A'_{i,k} \cdot T_{i,k} \tag{14}$$

Matrix $A'_{i,k}$ is a 3×3 matrix, each element of which is a function of the rotation parameters. Expression (14) has to be minimal and has the form of a quadratic in terms of vector $T_{i,k}$. The translation vector $T_{i,k}$ that minimizes the quadratic (14) is the eigenvector of $A'_{i,k}$ that has the smallest eigenvalue. The actual minimal value of (14) is this eigenvalue which, since $A'_{i,k}$ is a function of the rotation parameters, is also a function of the rotation parameters. This way the six-dimensional problem of minimizing (13) is reduced to a three-dimensional problem of minimizing the smallest eigenvalue of $A'_{i,k}$ which depends on three parameters.

2.4.3. Introducing More Frames

It is easier, now that we have slashed the dimensionality of the two-frame problem, to proceed to the many-frame case.

We define the overall translation vector $\mathbf{T}_1$ that is composed from the vectors $T_{i,i+1}$, $i = 1 \cdots N-1$ one on top of the other,

$$\mathbf{T}_1 = \left[T_{1,2,x},\ T_{1,2,y},\ T_{1,2,z},\ T_{2,3,x},\ T_{2,3,y},\ T_{2,3,z},\ \ldots,\ T_{N-1,N,z} \right]^T$$

which alternatively we can write as

$$\mathbf{T}_1 = \left[T_{1,2},\ T_{2,3},\ \ldots,\ T_{N-1,N} \right]^T$$

meaning a $3 \cdot (N-1)$-dimensional vector.

In the same way the matrix $\mathbf{R}_1$ can be defined as a $3(N-1) \times 3(N-1)$ matrix of the form

$$\mathbf{R}_1 = \begin{bmatrix} R_{1,2} & & & \\ & R_{2,3} & & \\ & & & \\ & & & R_{N-1,N} \end{bmatrix}$$

where the entries $R_{1,2}$, $R_{2,3} \cdots$ are 3×3 blocks filled by the corresponding matrices. Finally we define the shift matrix S as

$$S = \begin{bmatrix} 0 & I & & & \\ & 0 & I & & \\ & & 0 & & \\ & & & & \\ & & & & I \\ & & & & 0 \end{bmatrix}$$

where I is a 3×3 identity matrix and the blocks are filled in as before. As before, S is a $3(N-1) \times 3(N-1)$ matrix. Of course, $S^{N-1} = 0$.

As can be seen from the previous definitions $\mathbf{R}_1$, $\mathbf{T}_1$ represent the rotations and translations between each of the $N-1$ frames and its next frame, e.g. between the first and the second, between the second and the third, etc.

This is enough information to describe the whole motion because using it we can express the motion between any pair of frames.

Since we have $R_{i,i+k} = R_{i+k-1,i+k} \cdot R_{i,i+k-1}$ and $T_{i,i+k} = R_{i+k-1,i+k} \cdot T_{i,i+k-1} + T_{i+k-1,i+k}$, it is trivial to prove that

$$\mathbf{T}_k = S^{k-1} \left\{ \sum_{i=0}^{k-1} (\mathbf{R}_1 S^T)^i \right\} \cdot \mathbf{T}_1 = \mathbf{L}_k \cdot \mathbf{T}_1 \tag{15}$$

and

$$\mathbf{R}_k = S^{k-1} (\mathbf{R}_1 S^T)^{k-1} \cdot \mathbf{R}_1 \tag{16}$$

The same way we constructed the $\mathbf{R}_i$'s, we can construct the $\mathbf{A}_i$'s so that

$$\mathbf{A}_i = \begin{bmatrix} A'_{1,i+1} & & & & \\ & A'_{2,i+2} & & & \\ & & A'_{N-i,N} & & \\ & & & 0 & \\ & & & & 0 \end{bmatrix}$$

Since for $i > 1$ we have less than $N-1$ of these matrices $\mathbf{A}_i$ is supplemented by $i-1$ 3×3 blocks of zeros.

The expression

$$\mathbf{T}_1^T \cdot \mathbf{A}_1 \cdot \mathbf{T}_1 \tag{17}$$

represents the sum over all points of the squares of the common normals of pairs of lines from consecutive frames. In Figure 8 there is an example for four frames. The image point vectors p_{13}, p_{23}, p_{33}, p_{43} all meet at (or rather pass close to) P_3. In the circle in the right of the figure the common normals of the pairs of lines in consecutive frames are represented by single lines. The sum of squares of these normals for all points is given by (17). The reason for this is that

$$\mathbf{T}_1^T \cdot \mathbf{A}_1 \cdot \mathbf{T}_1 = \sum_{i=1}^{N-1} Ti_1 T \cdot A_{i,i+1}{}^! \cdot Ti_1$$

As we showed in the previous section, $\sum_{i=1}^{N-1} Ti_1 T \cdot A_{i,i+1} \cdot Ti_1$ is the sum of the squares of the length of one common normal per point, for all points. This is $N-1$ terms, one for each pair of consecutive frames. The expression

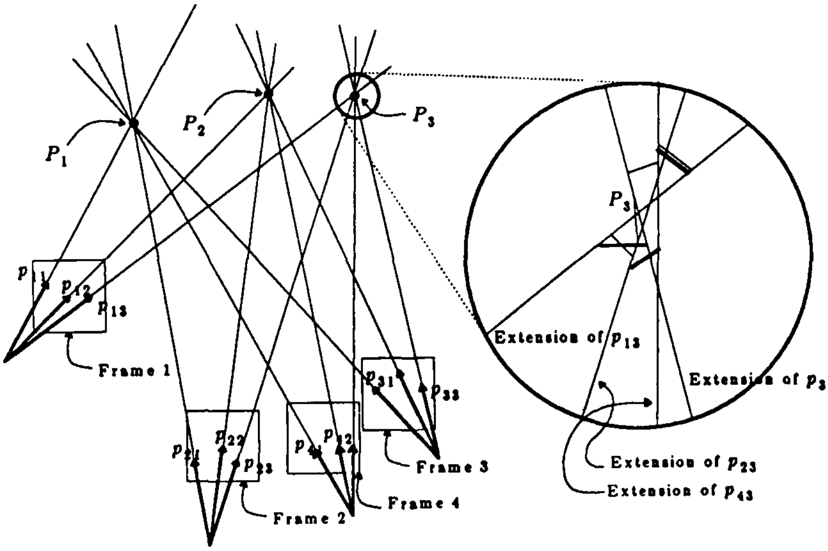


Figure 8. An example with four frames. Single lines show the first term of summation (19); double lines indicate the second, etc.

$$\mathbf{T}_k^T \cdot \mathbf{A}_k \cdot \mathbf{T}_k \qquad (18)$$

gives $(N-k)$ terms that represent the sum of the squares of the common normals of pairs of lines from frames $(1, 1+k)$, $(2, 2+k)$ $\cdots$ etc. In Figure 8, for $k = 2, 3$ these normals are the double and triple lines. The reason is the same as before. Making use of (16) we see that (18) can be written as

$$\sum_{k=1}^{N-1} \mathbf{T}_1^T \cdot \mathbf{L}_k^T \cdot \mathbf{A}_k \cdot \mathbf{L}_k \cdot \mathbf{T}_1 = \mathbf{T}_1^T \cdot \mathbf{B} \cdot \mathbf{T}_1 \qquad (19)$$

$\mathbf{B}$ is a $3(N-1) \times 3(N-1)$ matrix; otherwise the same holds as for $A'_{i,k}$ of eq. (14) in the previous section. It is a function of a set of motion parameters and its smallest eigenvalue, which is also a function of the same parameters, is the smallest value expression (19) can take. So the same comments apply about minimizing this eigenvalue, and this can be done efficiently.

One technique we can use to speed up evaluation is to "align" all the frames so that the guesses for the rotation matrices are identity matrices. We can do that by rotating the axes of $A_{i,k}$ defined in eq. (13) so everything is aligned to the last frame. Matrix A_{ij} depends on the set of correspondence pairs that belong to frames i and j and aligning it to the last frame requires rotation of every image point vector in frame i by R_{iN} and every image point vector in frame j by R_{jN}. We describe how one can rotate them all at once, thus saving computation time.

We define the 9×9 rotation matrix $R_{a_{ij}}$ (a is not an index; the same for b in the next definition)

$$R_{a_{ij}} = \begin{bmatrix} r_1 & 0 & 0 & r_2 & 0 & 0 & r_3 & 0 & 0 \\ 0 & r_1 & 0 & 0 & r_2 & 0 & 0 & r_3 & 0 \\ 0 & 0 & r_1 & 0 & 0 & r_2 & 0 & 0 & r_3 \\ r_4 & 0 & 0 & r_5 & 0 & 0 & r_6 & 0 & 0 \\ 0 & r_4 & 0 & 0 & r_5 & 0 & 0 & r_6 & 0 \\ 0 & 0 & r_4 & 0 & 0 & r_5 & 0 & 0 & r_6 \\ r_7 & 0 & 0 & r_8 & 0 & 0 & r_9 & 0 & 0 \\ 0 & r_7 & 0 & 0 & r_8 & 0 & 0 & r_9 & 0 \\ 0 & 0 & r_7 & 0 & 0 & r_8 & 0 & 0 & r_9 \end{bmatrix}_{ij} \quad \text{and} \quad R_{b_{ij}} = \begin{bmatrix} r_1 & r_4 & r_7 & & & & & & \\ r_2 & r_5 & r_8 & & & & & & \\ r_3 & r_5 & r_9 & & & & & & \\ & & & r_1 & r_4 & r_7 & & & \\ & & & r_2 & r_5 & r_8 & & & \\ & & & r_3 & r_6 & r_9 & & & \\ & & & & & & r_1 & r_4 & r_7 \\ & & & & & & r_2 & r_5 & r_8 \\ & & & & & & r_3 & r_6 & r_9 \end{bmatrix}_{ij}$$

Notice that $A_{ij} = \sum_k a_{ikj} \cdot a_{ikj}^T$ and a_{ikj} is a 9×1 vector that depends on the images of the k^{th} point projected on frames i, j. Then $R_{aiN} \cdot a_{ikj}$ is the same vector as a_{ijk} but with frame i aligned with the last frame. Similarly, $R_{bjN} \cdot a_{ikj}$ is the same vector as a_{ijk} but with frame j aligned with the last frame. Taking into account that $R_{bjN} \cdot R_{aiN} \cdot a_{ikj} = R_{aiN} \cdot R_{bjN} \cdot a_{ikj}$ it is easy to see that

$$R_{aiN} \cdot R_{bjN} \cdot A_{ij} \cdot R_{bjN}^T \cdot R_{aiN}^T \qquad (20)$$

is A_{ij} aligned with the last frame. This alignment depends on $N-1$ rotation matrices $R_{1,N}$, $R_{2,N}$, ..., $R_{N-1,N}$ which are the building blocks for R_{aiN} and R_{bjN} i, $j = 1..N-1$. If we treat these as unknowns, then taking the first derivative becomes easy. First we show how to take the derivative of the matrix $\mathbf{B}$ and then from this the derivative of its smallest eigenvalue.

Matrix $\mathbf{B}$ is a sum of matrices as indicated by (19): $\mathbf{B} = \sum_{k=1}^{N-1} \mathbf{L}_k^T \cdot \mathbf{A}_k \cdot \mathbf{L}_k$ But matrices $\mathbf{L}_k$ are constant matrices now because they are functions of the rotation matrix $\mathbf{R}_1$ which is now set equal to identity matrix. Only $\mathbf{A}_k$ is a function of the rotation parameters as seen from expressions (20), (14) and the definition of $\mathbf{A}_k$.

Now we are ready to find the derivative of $\mathbf{B}$. The derivative with respect to each one of the $3(N-1)$ Rodrigues parameters[18] (three for every $R_{1N}, \cdots, R_{N-1,N}$) is a $3(N-1) \times 3(N-1)$ matrix. To find it first we have to find the derivatives of the rotation matrices R_{iN}. For a rotation matrix R its derivative with respect to its j^{th} ($j = 1, 2, 3$) Rodrigues parameter is $R^{(j)} = \frac{1}{2}(R + I) B_j (R + I)$ where

$$B_j = \begin{bmatrix} 0 & -\delta_{3j} & \delta_{2j} \\ \delta_{3j} & 0 & -\delta_{1j} \\ -\delta_{2j} & \delta_{1j} & 0 \end{bmatrix}$$

After computing the derivatives of R_{iN}, $i = 1 \cdots N-1$ we can trivially find the derivatives R_{aiN}, R_{bjN}

[18] If the rotation matrix R corresponds to quaternion $q = (1, b_1, b_2, b_3)$ then the Rodrigues parameters are b_1, b_2, b_3.

$i, j = 1..N\text{-}1$, which is just shifting and replicating the elements of the corresponding matrices. From this we can find the derivatives of expression (20) and from then on the procedure is the same as for evaluating the matrix **B**. If $\mathbf{B}^{(i)}$ is the derivative of **B** with respect to the i^{th} parameter then the derivative of the smallest eigenvalue is

$$\lambda_o^{(i)} = \phi_o^T \cdot \mathbf{B}^{(i)} \cdot \phi_o$$

where ϕ_o is the eigenvector of **B** that corresponds to the smallest eigenvalue. If we find the derivatives with respect to all the parameters we can form the vector $\nabla\mathbf{B}$ which is the direction of the steepest increase of the eigenvalue. If $\mathbf{B}'$ and $\mathbf{B}''$ are the first and second derivatives of **B** along this direction then the second derivative of the eigenvalue along the same direction is

$$\lambda'' = \phi_o^T \cdot \mathbf{B}'' \cdot \varphi_o + \sum_{i=1}^{3N-4} \frac{(\phi_i^T \cdot \mathbf{B}' \cdot \phi_o^2)}{\lambda_o - \lambda_i}$$

where ϕ_i and λ_i are the eigenvector eigenvalue pairs of $\mathbf{B}$.[19]

2.4.4. Experiments

We ran a set of comparative experiments on synthetic images to test this algorithm. A synthetic "object", 20 f's away and 5 f's in its bigger dimension, was projected onto the image plane. (f is the focal length; it is one unit of length.) We used five frames and random motion between the frames. We passed the data through two two-frame algorithms [Tsai and Huang, 1984] and [Spetsakis and Aloimonos, 1988], and then through the algorithm just described. The results are shown in Figures 9, 10, and 11. The x-axis is the error in the input. The distance between the actual image point and the one used as input is expressed as a percentage of f. The output is expressed for the translation as the sine of the angle between the actual translation and the estimated one and the average over the four interframe motions is displayed. For rotation the sine of the angle between the two axes of rotation (actual and estimated) is displayed in the same way. The difference in rotation angles follows a pattern similar to the sine of the angle of the two axes and it is not displayed (it would make the diagram messy). It is obvious from the diagrams that there is an improvement of almost two orders of magnitude.

From the preceding analysis, that was basically an example from our research on the stability of visual computations, the following conclusions can be drawn:

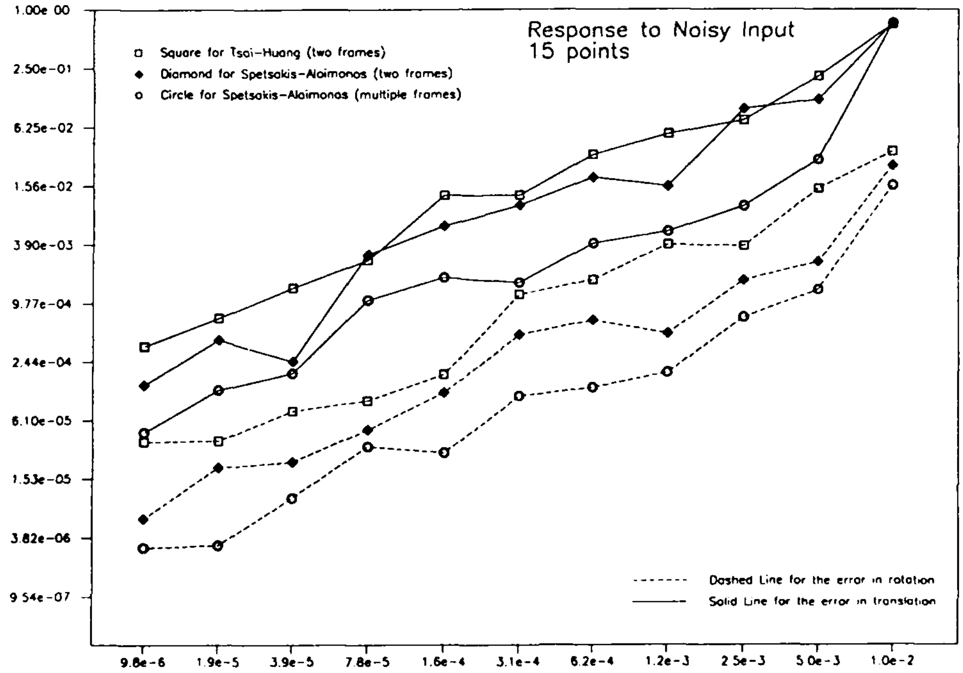


Figure 9.

[19] The second directional derivative of **B** can be found the same way but it costs $3(N\text{-}1)$ times more computation time than the first derivative. Since we have the first derivative accurately it is much cheaper to find it numerically.

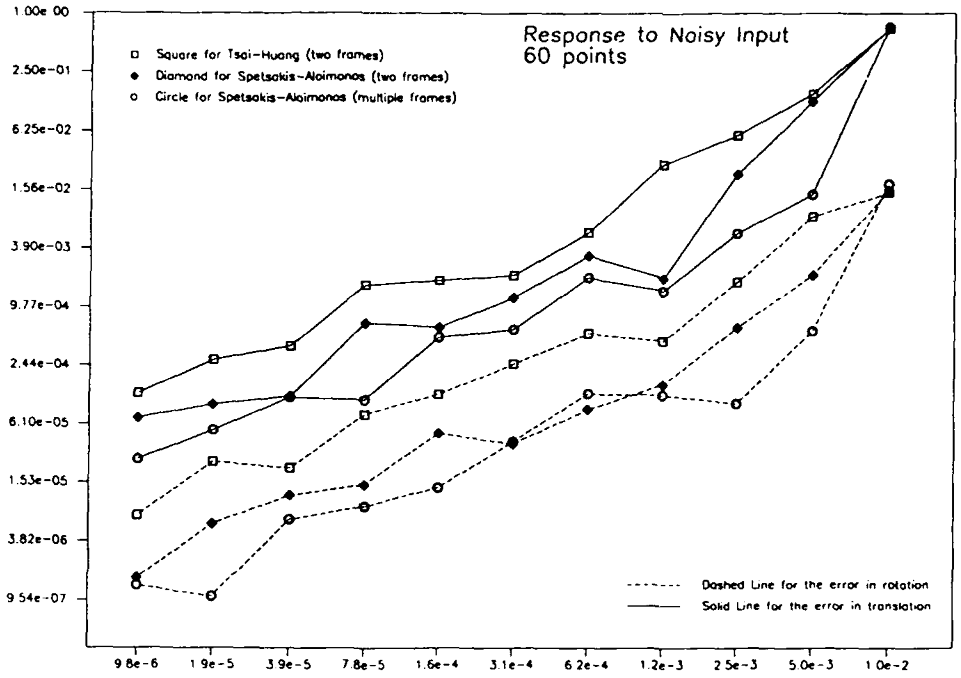


Figure 10.

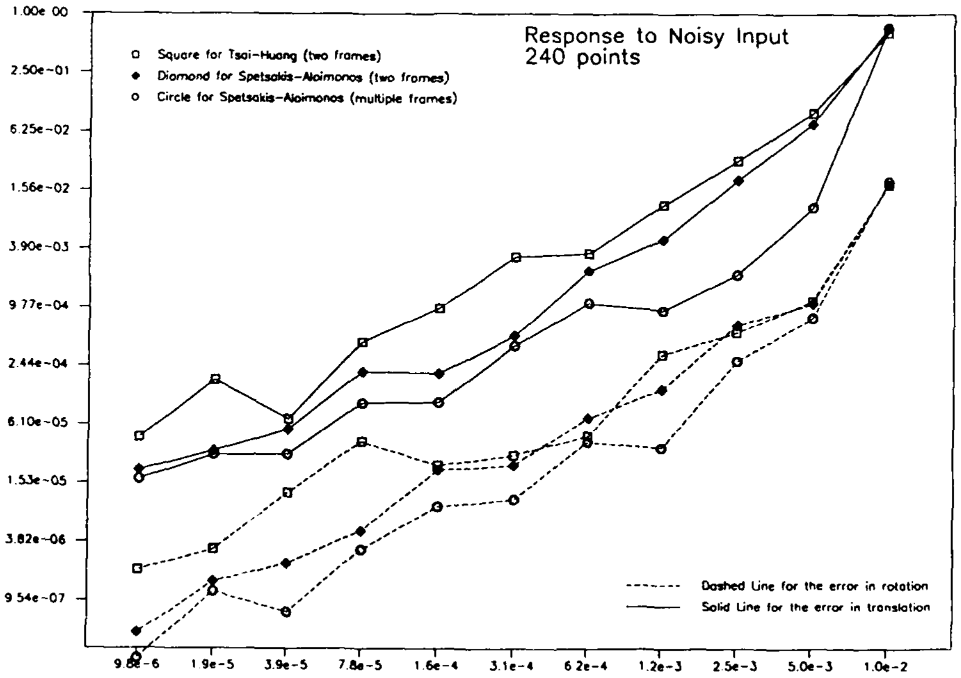


Figure 11.

- Optic flow (correspondence in two frames) cannot accurately estimate structure in the presence of noise, due to difficulties inherent in the problem itself.
- Redundancy (use of multiple frames), which is a well understood technique, used in several engineering problems for dealing with noise, seems to be very promising in increasing the stability of structure and motion estimation from dynamic imagery.

This concludes the description of our research in the analysis of specific visual modules. For analysis of various different modules the interested reader is referred to [Aloimonos, 1988]. The next section proceeds with the problems of module unification and integration.

3. MODULE UNIFICATION AND INTEGRATION

While our research on individual modules continues (for the discovery of optimal estimators and stability analyses), it has become clear that most modules are faced with the solution of ill-posed problems. So, we should proceed with the integration step, i.e., combine information from various cues in order to make the reconstruction problems well-posed and their solutions robust. But what should our methodology be in trying to understand how to combine information from different cues (coupling of modules)?

Not surprisingly, the general principles of top-down and bottom-up discussed in the first section apply here as well. A top-down approach would consist of a general theory of combining information from different visual cues and then if we wish to combine texture and contour, motion and stereo, shading and motion, and so on, in order to achieve unique and robust solutions to visual reconstructions, we would just have to apply this general framework to the specific coupling at hand.

A bottom-up approach would consist of considering several individual problems, i.e. combining shading and motion, contour and motion, texture and motion, shading and stereo, contour–texture and motion, and so on, and solving them individually and separately, with the hope that if we consider all possible combinations of cues, then we will have a complete theory of integrating visual modules. At this stage there is no evidence for preferring one approach over the other.

In [Aloimonos and Basu, 1988; Aloimonos, 1986] we presented an outline of the bottom-up approach. Here I describe a top-down approach, a general theory for combining information in vision. This is the theory of regularization augmented in order to efficiently deal with discontinuities, and hereafter called discontinuous regularization. The analysis follows [Shulman and Aloimonos, 1988].

This theory of discontinuous regularization can be used not only for module integration but it can also serve as a unified theory for solving early vision problems such as edge detection, interpolation, surface reconstruction, stereo, optic flow computation, shape from shading, shape from texture, patterns, motion, etc. It has been suggested [Poggio et al., 1984] that standard regularization [Tichonov and Arsenin, 1977] can serve as a unifying theory for early vision processes. However, some of the characteristics of the algorithms that result from standard regularization are the following:

(a) The algorithms are based on models that describe the relationship between the desired variables and the image measurements. These models contain parameters which are usually determined in an ad hoc way or through experimentation. In other words, the modeling process needs to make some assumptions which both restrict the applicability of the model and result in parameters whose value is to be determined before the algorithm can be used; and no systematic way of computing these parameters has been described.

(b) The algorithms do not improve with experience. That is, the algorithms are not equipped with the necessary machinery so that they can improve themselves automatically (learn) from examples of past work.

(c) The algorithms fail when the quantity to be computed is a discontinuous function (or nondifferentiable, i.e., has corners).

3.1. UNIFICATION

Poggio et al. [1984] showed that the theory of regularization can serve as the basis for the computational theories behind early vision processes. Indeed, in all the above processes, we have to deal with noisy data and the constraints relating data with real world variables are only approximately correct. Poggio et al. [1986] has suggested using the theory of Tikhonov stabilizers to regularize ill-posed vision problems such as the above. That means we use a priori information about the smoothness of the solution to determine a unique numerically stable solution. If $L(P) = 0$ is the constraint relating image data to the value of the solution ω at point P, we define a smoothness measure $S(P)$, which is usually a linear combination of derivatives of the real-world variable ω and our solution is the ω that minimizes $\| L \|^2_{N_1} + \lambda \|S\|^2_{N_2}$ where the $\| \; \|_{N_i}$ are norms and λ is a parameter measuring relative importance of consistency with the constraint $L = 0$ and the smoothness condition $S = 0$. The norms used are usually the L^2 norm $\left[\int f^2\right]$. This approach to regularization we will call the standard theory.

Another procedure is to minimize $\| L \|_{N_1}$ subject to a limit on the total amount of smoothness, $\|S\|_{N_2}$, which must be less than or equal to some number $S_{\max}$, or to minimize $\|S\|_{N_2}$ subject to $\| L \|_{N_1} \leq L_{\max}$. Both techniques are equivalent to minimizing $\| L \|^2_{N_1} + \lambda \|S\|^2_{N_2}$ for appropriately chosen λ.[20] However, the difficulty with regularization is that it tends to smooth over discontinuities and the visual world is rich in discontinuities. If we reduce the value of λ, we do less smoothing over discontinuities but we obtain a less noise-robust solution. If we use a second derivative smoothness term, there can be less smoothing over discontinuities but we introduce oscillation not present in the real world or data.

To summarize, regularization can serve as a unifying theory for early vision processes, if it can deal efficiently with discontinuities and if there is a systematic way for finding what the smoothness term should be or what the value of the parameters involved (for example λ) should be.

For a clear description of the ill-posed nature of early vision problems, see [Poggio et al., 1986]. The next section discusses previous work on discontinuous regularization; we denote the constraint of a particular problem by $L(\omega) = 0$, where ω is the vector of the desired parameters.

3.1.2. Previous Work on Discontinuous Regularization

One approach to discontinuous regularization is to first segment the image into homogenous regions and then to regularize within each region. Yachida [1983] studying optical flow does not smooth over regions where the variance of the first derivative of intensity (i.e. the data, the coefficients in the constraint $L = Au + Bu + C = 0$ on flow u, v) is large. Schunk [1983] iteratively segments the flow field and regularizes within each region. However, segmentation is not a solved problem. One of the purposes of knowing ω is facilitation of segmentation.

We can also proceed by dividing the set of points into boundary and non-boundary points. Assume there is a known probability that a random point be a boundary point and at such points all values of S are equally likely. At other points, the usual variational condition is acceptable but we do not excessively penalize large S (large S means probable discontinuity). Thus we minimize $\int \left[L^2 + \lambda\, g_T(S)\right]$ where $g_T(S) = \text{minimum}\,(S^2, T^2)$. T is a threshold depending on the fraction of points that are discontinuities. Geman and Geman [1984] studied the case where ω has discrete range (e.g. ω is a binary function) and the domain is a discrete lattice. Marroquin [1984, 1985] extends to the case where ω is real-valued and Mumford and Shah [1985] allow for a continuous domain. The variational conditions obtained are solved using Monte-Carlo techniques or by deterministic approximations to them such as the mean-field approximation. Blake and Zimmerman [1987] solve the condition by a continuation method known as "graduated non-convexity". The solution is generally not unique. *All methods* suggested for finding the solution either are not guaranteed to converge to a global minimum or cannot be proven reasonably efficient. Furthermore the assumption that all $S^2 > T^2$ are equally good is also questionable.[21]

Another discontinuous regularization theory that implicitly segments has been proposed by Terzopoulos [1986]. In his theory, S is a linear combination, $\lambda_1 S1 + \lambda_2 S2$ of first and second order derivatives and λ_1, λ_2 are

[20] Researchers who do not explicitly regularize ill-posed problems usually compute some sort of weighted average of the raw solution or simply do not compute the solution in regions of large noise. Instead of weighted averages, some form of non-linear smoothing may be used. In this case regularization is implicitly being performed.

[21] It is worth noting at this point the theory of convex regularization developed in our laboratory [Shulman and Hervé, 1989]. As emphasized, the idea that all large values of S are equally bad is questionable. We have computed many histograms of difference quotients of intensity; the tails are not particularly flat. In a crowded room, depth distances between occluding objects are not uniformly distributed. Too large a distance is unlikely because there is bound to be some other object in between two objects that are too far apart. The equal goodness assumption is saying in the case of a second derivative smoothness constraint on depth that the world roughly speaking consists of planar surfaces and sharp corners (discontinuities in orientation). Indoors, this assumption is often true, but an outdoor scene can often have more rounded than sharp corners. Actually we simply do not know what the best penalty function is, what g_T should be chosen. We do not know the probability distributions of the S_i. Let us assume that (1) the distribution is symmetric about 0; so for x, it is equally likely that S_i *equals* x *and that it equals* $-x$; (2) the distribution is a mixture so we can write $S = (1 - B)G + B\,H$ where B is a binary random variable taking the values 0 and 1 (not bad point, bad point), G is a Gaussian random variable, and H is a random variable with unknown probability density. The best we can do is choose a penalty function that minimizes solution error under the worst possible H. Thus we want the penalty function $P : P = \min_{P \text{ a penalty function}} \max_{H \text{ a distribution}}$ expected solution error (P, H). Looking for the minmax penalty function is the same as finding the worst, least informative H and choosing the P corresponding to that H. This least informative H is probably too uninformative. We are only given that H is symmetric; otherwise it is entirely unlimited. In practice we can learn additional constraints on H that are reasonably certain to hold. Furthermore we are assuming we know the expected value of B, we know the expected fraction of bad points. This is equivalent to knowing the threshold, T. In reality, finding a good T is nontrivial.

Huber [Huber] shows that the least favorable distribution, the one causing the greatest expected mean square error, is the distribution corresponding to the penalty function $g_T(x) = x^2$ for $|x| < T$, $= T^2 + 2T(|x| - T)$ for $|x| \geq T$. This function is convex; thus all local minima are global *minima*. If we add a small quadratic to g_T, we obtain a strictly convex function $g_{T_\epsilon}(x) = x^2$ for $|x| < T$, $= T^2 + 2T(|x| - T) + \epsilon(|x| - T)^2$ for $|x| \geq T$.

Now we are guaranteed unique local minima to our variational condition provided we have enough data points. In practice, we do not seem to need the extra term involving epsilon. If we use the expression, epsilon should be any small positive number. This approach gives reasonable results as Figures 12-13 indicate.

position dependent. Terzopoulos is primarily concerned with the case where λ_1, λ_2 are constant except at a small number of points where one or both are zero. A complex iterative procedure is used to determine these points. The regularization theories discussed above can be augmented to incorporate knowledge that boundary curves are smooth except at a few corner points or other knowledge about boundary shape. The basic difficulties with these techniques are not changed. Lee and Pavlidis [1987] use a discretized standard regularization to obtain an initial estimate of the solution. Then λ is set equal to 0 at points where $L^2 + \lambda S^2$ is large and standard regularization is performed once more.

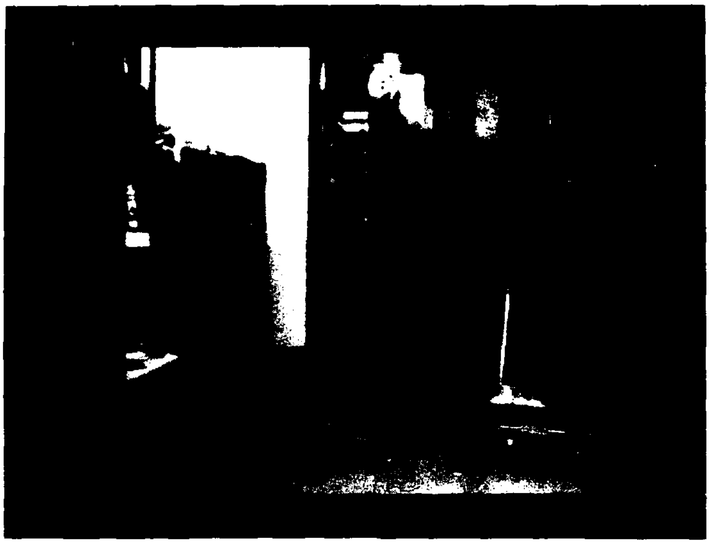


Figure 12. The lab scene used for the experiemnt It represents the image before the motion of the camera. The motion is horizontal.

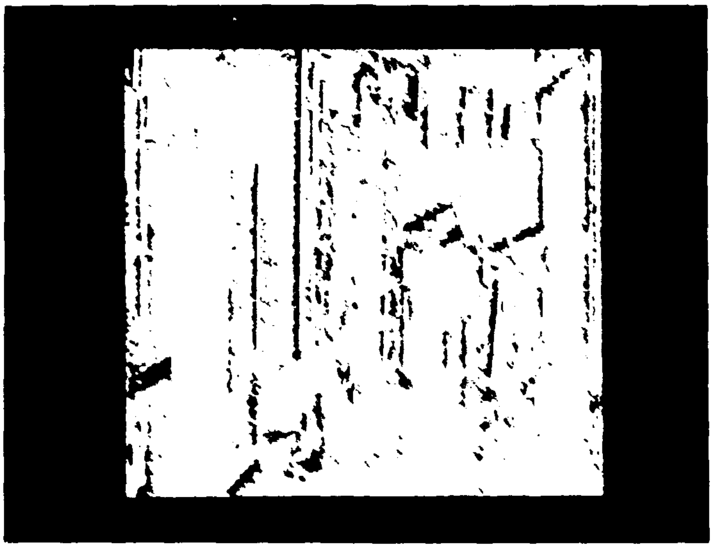


Figure 13. Discontinuity flow map Dark areas represent discontinuities in the flow. The reader can observe edges in Fig 12 (discontinuities in image intensity) that do not correspond to flow discontinuities in Fig 13

Theories that do not make a rigid binary distinction between boundary and non-boundary have been applied to optical flow estimation. Such theories are needed because there may be points where flow is not particularly smooth but jumps in flow are not sharp enough to be true discontinuities. We do not wish to oversmooth the values of ω in such a region. Haralick [Haralick and Lee, 1983] in applying the facet model (i.e. piecewise polynomial model) to optical flow suggests that in addition to the flow equation $L = \partial E/\partial x \; \partial x / \partial t + \partial E / \partial y \; \partial y / \partial t + \partial E / \partial t = 0$, we also require the derivatives of L be zero.

Nagel and Enkelmann [1986] used "oriented smoothness" to insure that more smoothing is done in the direction of the gradient $(\partial E / \partial x, \partial E / \partial y)$ of intensity than in the orthogonal direction and that we smooth more where the gradient is small and the flow equation most vulnerable to noise. Writing $\vec{u} = (\partial x / \partial t, \partial y / \partial t)$ for the flow vector and $\nabla \vec{A} = \begin{pmatrix} \frac{\partial A_1}{\partial x} & \frac{\partial A_1}{\partial y} \\ \frac{\partial A_2}{\partial x} & \frac{\partial A_2}{\partial y} \end{pmatrix}$ for the gradient of vector $\vec{A} = (A_1, A_2)$, they use the smoothness measure $\lambda \,\mathrm{trace} \int \left[\nabla \vec{u} \left(\nabla E \,(\nabla E)^T + b \,\nabla\nabla E \,(\nabla\nabla E)^T \right)^{-1} \nabla \vec{u}^T \right]$ instead of the measure $\int \lambda \,\mathrm{trace}\, [\nabla \vec{u} \,(\nabla \vec{u})^T]$ suggested by Horn-Schunk. Here b is a constant that needs to be determined. The resulting Euler-Lagrange equation is linear in the unknowns. No rigid binary distinction of boundary from non-boundary is made. But the variational condition is not completely motivated (several similar conditions are suggested under the name "oriented smoothness"). The theory is only applied to flow and not to other problems such as finding derivatives like $\partial E / \partial x$. Furthermore, should not the value of E_t, E_{tx} or even E_{xxx}, if we can estimate these derivatives accurately, influence the amount of smoothing we do?

We propose a general theory of linear discontinuous regularization to address these issues. We retain the preservation of sharp changes in ω that are not boundaries. The basic insight of the theory is that errors and smoothness measures of nearby points are correlated so we need to add terms involving the derivatives of S and L to the variational condition. In addition, the parameters of the theory are learned through adaptive estimation.[22] Also, this theory does not make a rigid binary distinction between discontinuity and non-discontinuity points.

3.1.3. Colored Noise

Something strange happens at discontinuities when we apply standard regularization theory. For simplicity consider the problem $A\omega = B +$ noise where $A = 1$ at a small number of points (at these points we have data) and 0 elsewhere. The variable ω is one-dimensional and the preferred smoothness measure is $\lambda \int (d\omega/dx)^2$ where x is the coordinate of a one-dimensional image. Examine Figure 14.

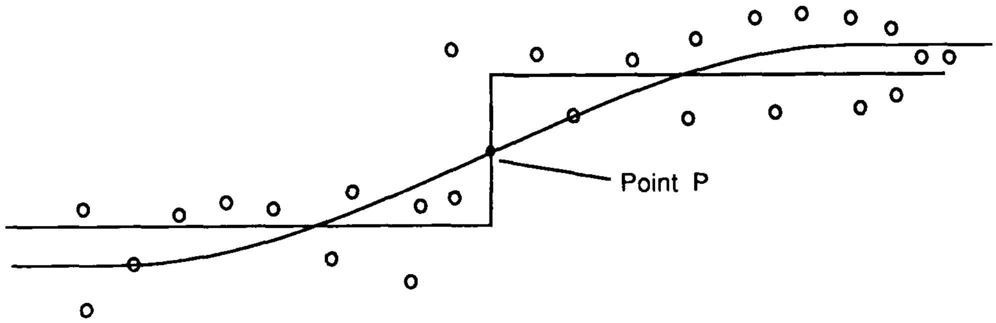


Figure 14.

The original signal is an edge at point P, the circles are data points, the curved line is the usual regularized solution. We note that immediately to the left of P there is a large positive difference between solution and data and to the right there is a large negative difference. Now if we consider a signal with many discontinuities or many signals with discontinuities, we find that there is a negative correlation between the errors of points each of which is a short distance δ from a discontinuity and which are on opposite sides of the discontinuity. The errors of nearby points can have negative correlation just by chance. But *it is curious* that this should happen just where the discontinuities are if these discontinuities are genuine discontinuities resulting from material change or other abrupt transitions in the nature of the object being viewed. Thus it seems it might be worthwhile to study the correlation matrix of the errors. Negative correlation is a sign that discontinuities are being smoothed. We can try to alleviate this using a different smoothness measure such as $\lambda \int (d^2\omega/dx^2)^2$ but this just gives us a Gibbs phenomenon—oscillations not found in the original signal are found in the solution. We can try to reduce λ but

[22] There are various other ways with which the parameters can be estimated, but we have only implemented learning.

this makes our solution more vulnerable to noise. We could try to let λ vary with position but how do we do this?

The basic problem is this: the regularization is an example of a least squares solution. Least squares can be justified if we are dealing with Gaussian probability distributions, i.e. if the distribution of $d\omega/dx$ is Gaussian. But a quick look at a diagram in Feller [1966] will show that a Gaussian random walk almost certainly has no discontinuities while the random walk based on the Cauchy distribution has many discontinuities. A quadratic smoothness measure does not believe in discontinuities. It overpenalizes big jumps in ω. The only remedy is to overpenalize a proposed solution $\hat{\omega}$ that does not reflect evidence in the data that a jump exists. This suggests it might be interesting to study the derivatives of the error. A similar derivation will show the relation between correlation and the derivative of the error. Variance (error at P - error at R) = variance (error at P) + variance (error at Q) - 2 (covariance of error at P with error at Q). This is just because $E\left((a-b)^2\right) = E(a^2) + E(b^2) - 2E(ab)$ where E means expectation. Thus if nearby points have correlated errors, the size of the difference between these errors tends to be small and in fact derivatives of the error tend to be small.

3.1.4. Discontinuous regularization

In this section we show that we can recover real world variables (or intrinsic ones), i.e. solve ill-posed early vision problems, by minimizing an expression involving not only the constraint plus smoothness but a combination of the derivatives of the constraint and the derivatives of the smoothness.

The usual regularization theory has a *simple statistical interpretation*. Minimizing "energy" = $\int (L^2 + \lambda S^2)$ is the same as maximizing $e^{-k\text{ energy}}$. The expression $e^{-k\text{ energy}}$ can be regarded as a probability if k is chosen correctly. The constant k is essentially an irrelevant normalizing factor. Standard regularization theory maximizes the likelihood of the solution ω if λS and L are Gaussian, equal variance, independent random variables and the "smoothness", S, of distinct points and the "error", L of distinct points are independent also. If S and L are not Gaussian, standard regularization theory provides the best linear equation for ω if the λS, L are equal variance, uncorrelated and errors[23] and smoothness of different points are uncorrelated. This is essentially the Gauss-Markov theorem [Rao, 1973]. But the uncorrelatedness assumption is often false. If var, covar represent variance, covariance respectively, then $\text{var}(A-B) = \text{var}(A) + \text{var}(B) - 2\text{covar}(A,B)$. If $L(P)$ and $L(P+dP)$ are highly correlated, *dL will tend to be small.* So instead of minimizing $\int [L^2 + \lambda S^2]$, we minimize (in the 1-dimensional case) $\int \left[\sum_0^\infty a_i (D^i L)^2 + \sum_0^\infty b_i (D^i S)^2\right]$. D^i means i^{th} derivative and the parameters a_i, b_i need to be determined. The generalization to two or more dimensions should be clear.

Let us now mention two additional justifications for our discontinuous regularization theory, two additional reasons $D^i L$ should be small. One reason is that we may care more about discontinuities in ω than about the value of ω. So we really want to know the values of $D^i\omega$. If $L = A\omega - B$ then $DL = (DA)\omega + A(D\omega) - DB$. If DA is small, then $A(D\omega) - DB = 0$ is a simple direct relation between data and the derivative of the solution. In general the relation is more complex, but $DL = 0$ directly relates jumps in the data with jumps in ω. The other reason for constraining derivatives of the error is that a quadratic smoothness measure such as S_2 overpenalizes large jumps in ω. We need the quadratic smoothness measure to guarantee the Euler-Lagrange equations obtained are linear in S. So to compensate for an incorrect smoothness measure, we need to use an incorrect data consistency measure L'^2. The crucial advantage of this theory is that if L, S are linear in ω, then the Euler-Lagrange equations we need to solve are linear in ω and the coefficients are locally computable.

Another way to think of discontinuous regularization is that not only must the solution be consistent with the data but the derivatives of the solution must also be consistent with the derivatives of the data. Of course, there is a trade off: if we want to increase our sensitivity to shape discontinuities, for example, we have to decrease the accuracy with which we satisfy one (or both) of the other two constraints (i.e. that L be small and that the surface be smooth). So our determination of shape will be somewhat less accurate and since $L' = 0$ is not a perfect constraint, we risk preserving discontinuities in shape for which the data provide strong evidence even though these discontinuities are spurious and the data discontinuities really represent noise.[24]

[23] By error, we mean deviation from the constraint.

[24] There is another justification also, but due to lack of space we omit it. Standard regularization theory used ordinary least squares. Ordinary least squares can be justified *under the assumption* that we are dealing with independent Gaussian errors. But even if the errors are non-Gaussian, we can seek the best linear unbiased estimate of the real world variable ω. The best estimate means the estimate that minimizes the expected value of $\| \hat{\omega} - \omega \|_2^2$, where $\hat{\omega}$ is the estimated value of ω and ω is the true value of ω and $\| \; \|_2$ is the usual L^2 norm. We want our estimate to be unbiased so the expected value of $(\hat{\omega} - \omega)$ is 0. Linear means we can obtain $\hat{\omega}$ from a linear equation. More precisely we have equations of the form $\begin{matrix} A\omega = B + \text{error} \\ S\omega = 0 + \text{error} \end{matrix}$, where A is a linear operator, $A\omega = B$ is our data con-

In the sequel, in order to be able to explain things in a clear way, we concentrate on the shape from X problems, i.e. for every point (x, y) on the image plane (a unit square), we have a constraint $L(f, g, x, y) = 0$ that relates the shape (f, g) of the surface patch whose image is the point (x, y), with data measurements. The rest of the problems are addressed in a similar or analogous way.[25] Thus, in general a variational condition in discontinuous regularization theory takes the form:

$$\text{minimize} \int \left[\sum_0^\infty \sum_0^\infty a_{ij} (\partial^{i+j} L / \partial x^i \partial y^j) \right]^2 dx dy + \int \sum_0^\infty \sum_0^\infty b_{ij} (\partial^{i+j} f / \partial x^i \partial y^j)^2 + \int \sum_0^\infty \sum_0^\infty c_{ij} (\partial^{i+j} g / \partial x^i \partial y^j)^2 .$$

Here the a_{ij}, b_{ij}, c_{ij} are constant parameters. The first integral involves a summation from 0 to ∞ because it is arbitrary to require only the first order derivatives of L to be small and not to impose a requirement on the higher order derivatives. Similarly, the smoothness condition (i.e. the second and third integrals) must also involve a summation from 0 to ∞. Some of the coefficients a_{ij}, b_{ij}, c_{ij} can be 0. Because of the noisiness of higher-order derivative estimates, the coefficients a_{ij} should rapidly approach 0 as $i, j \rightarrow \infty$. Clearly, a problem that remains is to find the values of the coefficients a_{ij}, b_{ij} and c_{ij}.

Discretizing the problem and having assumed that the image is a unit square (for simplicity and without loss of generality) and considering grid side $1/k$ we have as the unknown shape the vector

$$\xi = (f_{11}, f_{12}, \ldots, f_{1k}, f_{kk}, g_{11}, g_{12}, \ldots, g_{1k}, \ldots, g_{kk})^T$$

We further assume that we know how to compute derivatives of intensity (data).[26] Once we know how to do that we know how to compute the derivatives of the coefficients that appear in the formula for L and thus we can compute derivatives of L. We will obtain an Euler-Lagrange equation of the form

$$A(L) \frac{\partial A(L)}{\partial f} = -\Phi_1 \xi$$

$$A(L \frac{\partial A(L)}{\partial g} = -\Phi_2 \xi$$

where Φ_1, Φ_2 are matrices and $A(L)$ is a matrix obtained by applying a possibly non-linear functional to the matrix L.[27] In fact, $A(L)$ is a sum of the form $\sum a_{ij} A^{ij}(L)$ where the a_{ij} are constant coefficients and the $A^{ij}(L)$ are approximations to the derivatives $\partial^{i+j} L / \partial x^i \partial y^j$. Thus finally we obtain

$$\sum a_{ij}^2 A^{ij}(L) \partial A^{ij}(L) / \partial f = \Phi_1 \xi$$

$$\sum a_{ij}^2 A^{ij}(L) \partial A^{ij}(L) / \partial g = \Phi_2 \xi$$

with the $A^{ij}(L)$ being functions that we know how to compute. Write $\phi_{ij}(\xi)$ for the known function $\begin{bmatrix} A^{ij}(L) \partial A^{ij}(L) / \partial f \\ A^{ij}(L) \partial A^{ij}(L) / \partial g \end{bmatrix}$; then we have

sistency constraint, S is a linear operator that incorporates the smoothness constraint (so if the smoothness measure is $\lambda \int \left(\frac{\partial \omega}{\partial x} \right)^2$, then $S = \frac{\lambda \partial}{\partial x}$), and "error" measures the inexactness of the constraints. We want an estimate $\hat{\omega}$ of ω that is of the form $\hat{\omega} = C(B) + D(0)$ for some linear operators C, D. According to the Gauss-Markov theorem [Rao, 1973], the least squares solution is the best linear unbiased estimate if the errors are uncorrelated and have equal variances. But in general noise is not white; it is colored. We have to decorrelate the error in order to apply least squares. Decorrelation, in general, is a complex task. So we will have to make special assumptions in order to be able to decorrelate quickly. Under special assumptions about the blurring kernel we can show that standard regularization applied to the decorrelated signal results in a minimization of the above form. For details see [Shulman and Aloimonos, 1988].

[25] They are actually simpler.

[26] Computing derivatives of intensity is an ill-posed problem itself. The theory presented in this paper can be used to regularize it. So we assume that this step has been completed (we have learned how to compute derivatives of intensity). Or we can use Poggio's method for approximating derivatives of intensity. If we assume that we are able to do linear learning this is simple. $\partial E / \partial x$, $\partial E / \partial y$, etc. are linear functions of E so we have equations such as $\partial E / \partial x = AE$ and we have to learn the matrix A that approximates the derivative. That means a teacher gives us a set of examples $(\partial E_1 / \partial x, E_1)$ $(\partial E_2 / \partial x, E_2)$, $\ldots$, $(\partial E_m / \partial x, E_m)$ $\cdots$ and we have to learn the A t at best approximates a solution to the system of equations $\partial E_i / \partial x = AE_i$ for all i. A priori, we expect that A will be obtained by applying a smoothing filter (such as a Gaussian) to a weighted sum of difference coefficients $\Delta E / \Delta X$, $\Delta^2 E / \Delta X^2$, $\Delta^3 E / \Delta X^3$, etc. In other words, we approximate E by a piecewise polynomial function, smooth this function with a Gaussian and then take the derivative.

$$\sum a_{ij}^{2}\phi_{ij}(\xi) = \Phi\xi$$

where the a_{ij}^{2} and Φ have to be learned. Again we will impose additional constraints to insure that the values of $(a_{ij}^{2}.\Phi)$ are unique.

To summarize this section, the theory of discontinuous regularization requires us to impose constraints on the derivatives of the "error" L.[28] This leads to equations of the form $\sum a_{ij}^{2}\phi_{ij}(\xi) = \Phi\xi$ where we need to learn a_{ij}^{2}, Φ. More generally, we could let the a_{ij}^{2} be matrices rather than constants (thus a_{ij}^{2} is a function of position and the constraints that the derivatives of L be small could be relaxed at certain points). We must describe a procedure for learning the matrices a_{ij}^{2}, Φ that is simple, robust, and leads to a unique solution.

3.1.5. Learning the Parameters

Techniques for learning the solutions to equations have been developed by researchers in stochastic approximation.[29] Many of these techniques apply to linear as well as nonlinear equations. There are special characteristics that low-level vision problems have that make it possible for us to accelerate the convergence of these techniques. This is because we have *a priori* knowledge that the solution must lie in a restricted subset of all possible solutions. We are also obliged to find the solution within a constricted subspace, because we must represent the solution by a neural network of limited size with a limited number of connections, which cannot represent an arbitrary function or even an arbitrary matrix. Techniques have been developed for finding the best solution within a restricted subspace.

First, we must discuss what exactly we want to learn and why. What does it mean to learn a solution to

$$\sum a_{ij}^{2}\phi_{ij}(\xi) = -\Phi\xi \tag{21}$$

when there may be not a unique ξ that satisfies (2)? What does it mean to have learned Φ from examples when the examples might not suffice to determine Φ (even under the assumption that we know the matrices a_{ij}^{2}), or when they give contradictory evidence as to what Φ is? What additional constraints need we impose to insure uniqueness of a_{ij}^{2}, Φ?

To answer the last question first, we are biased in favor of sparseness. Thus we want a_{ij}^{2} to be near zero if i,j are large in order to avoid computing noisy higher order derivatives and a_{00} to be nearly the identity matrix. The problem here is we could multiply both sides of (2) by an arbitrary matrix G. If we want to insure uniqueness we have to put a constraint on one of the a_{ij}^{2} or on Φ. The constraint a_{ij}^{2} is small or the least squares constraint minimize $\| \sum a_{ij}^{2}\phi_{ij}(\xi) + \Phi\xi \|_{2}^{2}$ leads us to the result $a_{ij}^{2} = 0$, al i,j and $\Phi = 0$ which make (2) useless. So we fix a_{00} to be close as possible to the identity matrix. This also tends to make a_{00} sparse and save space. If $a_{00} = I$ and $a_{ij} = 0$ for $i + j > 0$ then we obtain the equation

$$L\,\partial L/\partial g = -\Phi_{1}\xi$$

$$L\,\partial L/\partial g = -\Phi_{2}\xi$$

which is the usual nondiscontinuous regularization condition. Rewrite $\sum a_{ij}^{2}\phi_{ij}(\xi) = -\Phi\xi$ as

$$\left(\sum \hat{a}_{ij}^{2}\phi_{ij}(\xi)\right) + \Phi\xi = -\phi_{00}(\xi) \tag{21 A}$$

where $\hat{a}_{ij} = a_{ij}$ - Identity Matrix if $i = j = 0$ and $\hat{a}_{ij}^{2} = a_{ij}$ otherwise. Thus the first term in the equation represents the discontinuous correction to the usual regularization condition and we want this term to be as small as possible. To make it easier to work with, rewrite (21 A) in the form

$$\Gamma\,\Xi = -\phi_{00}(\xi) \tag{21 B}$$

where Ξ is the vector

[27] Indeed, if we are given the value of a function h at a finite number of points $P_1, P_2, P_3 \cdots P_n$ and we know these values are corrupted by noise, then the optimal approximation *to the derivative of h need not* be a linear function of the values $h(P_1)$, $h(P_2) \cdots h(P_n)$. This is true despite the fact that $\partial h/\partial x$ is a linear functional of h.

[28] L should be zero. When it deviates from zero it represents what we call "error".

[29] [Kushner and Clark, 1978; Wasan, 1969], statistics [Bromberg, 1985; Albert, 1972], mathematical learning theory [Tsypkin, 1973; Herkenrath et al., 1983], computer science [Kohonen, 1977], and electrical engineering communities [Kalman; Ljung, 1977; Maybeck, 1979].

$$\left[\phi_{00}(\xi), \phi_{01}(\xi), \phi_{10}(\xi), \ldots, \phi_{mn}(\xi), \xi\right]^T$$

and Γ is a matrix to be learned. The same questions we ask about Φ, we now ask about Γ: what do we do if the examples give incomplete information or inconsistent information about Γ?

The Γ we choose is the least squares solution. Under reasonable and quite general statistical assumptions, the least squares solution is the most probable value for Γ. Computing the least squares solution involves calculating the Moore-Penrose inverse. This procedure is optimal in the sense that it produces as accurate an estimate of Γ as we can obtain from our limited set of examples, but it requires a neural net to store large matrices and make very complex calculations. So instead of searching for the optimal solution of (21 B) we can calculate the best *solution within a restricted subspace.* The procedure for calculating this restricted subspace solution is very similar to that for calculating the full space solution but the computational complexity is much less and the speed of convergence is much greater. Another technique is the Robbins-Monro procedure. This results in slower convergence but the calculations are very easy to implement on a neural net, because the number of long-distance connections is minimal. Whatever procedure we use, it may take too many examples to learn Γ. We use a priori knowledge to accelerate convergence. For example, we know that Γ varies smoothly as a function of position. The Γ we use is partly determined by examples and partly by a priori knowledge. It is a weighted sum of the a priori term and a term learned from examples. The weight of the a priori term gradually lessens. As we acquire more examples, the influence of a priori knowledge on our estimate of Γ decreases.

Let us review what learning (rather than stipulating a priori) the value of Γ will accomplish for us. Γ hides the regularization parameter λ. This parameter might be allowed to vary from place to place. (We might require a different amount of smoothing near the boundary than at the center of the visual field.) Our smoothing condition might involve first, second, or higher-order derivatives or some linear combination of derivatives of different orders. Γ weights the relative importance of the derivatives of L being small. This can vary with position. By varying Γ, we can take into account all these possibilities. We do not know which of these possibilities is correct. That is why we want a general learning theory to help us find the proper Γ (to find the least squares solution to (21 B)).

3.1.6. Solution

As far as finding Γ is concerned, it is irrelevant that $\phi_{00}(\xi)$ and Ξ are functions of ξ. We just treat (21 B) as a *linear equation to be solved for Γ. We will also ignore the minus sign in equation (21 B) in order to simplify* notation. We are given a set of learning examples $\left(\Xi_i, \phi(\xi_i)\right)$ and we want to solve the system of equations

$$\Gamma \Xi_i = \phi(\xi_i) \tag{22}$$

Even if a teacher tells us the exact value of the shape vector, ξ_i, and of the image data that determine $\phi_{00}(\xi_i)$ and Ξ_i, we cannot expect (22) to be a strict equality because the model we used to determine Γ was an oversimplification. We cannot possibly take all factors into account. If the error is a zero-mean, normally distributed variable, it makes sense to seek a solution in the sense of least squares:

$$\text{minimize } \sum_i \| \Gamma\, \Xi_i - \phi_{00}(\xi_i) \|_2^2 \tag{23}$$

Assuming normality of the error, then the least-squares solution to (23) is the most likely value for Γ. Even if the error $\Gamma\Xi_i - \phi_{00}(\xi_i)$ is not Gaussian we can under reasonable assumptions such as that the errors $\Gamma\Xi_i - \phi_{00}(\xi_i)$ have bounded variance and the errors of examples i and $i+N$ are statistically independent for large N, obtain a central limit theorem which says that if there are a large enough number of examples the least square solution will be the best.

In case there are many least-squares solutions, choose the Γ of minimum norm, $\|\Gamma\|_2$. This will help to make Γ sparse. The minimum-norm, least squares solution to $AX = I$ is called the Moore-Penrose pseudo-inverse, A^+.

Many techniques exist for computing pseudo-inverses; not all of them are suited to the case where the data arrive in a stream (a temporal succession of examples.) One method that is suitable is Greville's [Albert, 1972; Ben Israel, 1978].

Another technique we can use is the Robbins-Monro procedure. The Robbins-Monro procedure can be shown to converge with probability one to the least-squares solution to $\Gamma\xi = \phi(\xi)$ under assumptions much weaker than the requirement that the errors be Gaussian. For example we can require only that the errors have identical, independent distributions and then we can obtain a convergence proof. These conditions are still somewhat restrictive. There are extensions to cases where the errors are a moving average of exogenous variables or where the error depends on Ξ.[30]

Even if we have convergence, we need not have convergence to the correct solution if our sample examples are not representative of the naturally occurring shapes. The solution will instead converge to the least-squares solution to the artificial problem presented by the atypical examples. What we require is that the vector spaces of naturally occurring possible examples be spanned infinitely often. Thus, for every m, the rank of the matrix $(\Xi_m, \Xi_{m+1}, \ldots, \Xi_s, \cdots)$ must be n. And we also require that we can choose a function $f(i)$ such that for all small i, the shape vectors $\Xi_{f(i)}, \ldots, \Xi_{f(i+1)-1}$ span the vector space and $\sum a_{f(i)} = \infty$.

Even if we use the optimal formula for estimation of the least squares solution, convergence can be quite slow. The variance of our estimates depends on the trace of the gain matrix. Thus, our rate of learning is inversely proportional to a large number n [Tsypkin, 1973].

The result of this slow convergence is that our estimate Γ may not be as sparse as it should be. Another possibility is that some of the values of $\Gamma = <\Gamma_{ij}>$ might be misleading. We might have a reasonably accurate estimate of each Γ_{ij} but a bad estimate of the high frequency components such as $\Gamma_{ij+1} - \Gamma_{ij}$. The estimate of these components might simply reflect noise. This may mean that although we can compute shape accurately, we *cannot accurately compute the difference in orientation of two nearby points.*

In other words, we still have to deal with the regularization problem. Each element of $\Gamma = <\Gamma_{ij}>$ reflects the relation between the value of some component of $\phi_{ab}(\xi)$ (or ξ itself) at some point $P_2(j)$ and the value of some component of $\phi_{00}(\xi)$ at some point $P_1(i)$. We say some component because, for example, $\phi_{ab}(\xi)$ at $P_2(j)$ has two components

$$A^{ab}(L)\partial A^{ab}(L)\ \partial f \quad \text{and}$$

$$A^{ab}(L)\partial A^{ab}(L)\ \partial g \quad \text{where} \quad A^{ab}(L)$$

is the approximation used for $\partial^{a+b} L \cdot \partial^a x \partial^b y$. Thus the matrix Γ has four parts: Γ_{ff}, Γ_{fg}, Γ_{gf}, Γ_{gg}. Γ_{ff} relates information about f to information about f ; Γ_{fg} relates information about f to information about g, etc. We expect all the difference quotients of the Γ will usually be small. That these difference quotients are small just says that the influence of the shape at some point P on the data at Q should be a smooth function of P and Q. So, regarding Γ as a continuous function, we will have a condition of the form

$$\text{minimize} \sum\sum(\Gamma_{ij} - \overline{\Gamma}_{ij})^2 + \sum\sum\lambda(P_1(i), P_2(j)) \cdot \left[(\Delta\Gamma/\Delta P_1(i))_k^2 + (\Delta\Gamma/\Delta P_2(j))_k^2 \right] \tag{24}$$

where $\overline{\Gamma}_{ij}$ is the value given by the recursive least-squares estimation of Γ. The sum is taken over all points in the grid. The subscript k means that if Γ_{ij} is a part of Γ_{fg}, then the difference quotient $\Delta\Gamma/\Delta P_1(i)$ only compares Γ_{ij} that relates $P_1(i)$ to $P_2(j)$ with other components of Γ_{fg} that relates points near to $P_1(i)$ to the point $P_2(j)$. We do not subtract components of Γ_{gf} from components of Γ_{fg} in computing these difference quotients. The parameter λ is a position dependent parameter which reflects our a priori knowledge about how Γ varies as a function of P_1, P_2 (i.e. our a priori probability distribution for $\Delta L/\Delta P_1$, $\Delta L/\Delta P_2$). This might seem ad hoc. The ad hocness here is not as bad as the ad hocness in assuming that the shape is smooth. Here we are only assuming that the function relating the shape to the data is smooth almost everywhere. Furthermore as we learn more and more examples, we can let $\lambda \rightarrow 0$. So, the influence of the ad hoc smoothing condition lessens. Polyak and Tsypkin [1980, 1979, Tsypkin, 1973] show that the choice $\lambda_s = \frac{1}{s}\tilde{\lambda}$ is optimal where λ_s is the parameter to use on the s^{th} example and $\tilde{\lambda}$ is the a priori value of λ. Of course, we could have chosen some other smoothing condition. As long as this other condition is quadratic, we will obtain a linear constraint in Γ. In general we will have a stabilizing family of smoothing conditions:

$$\text{minimize } \frac{1}{n^2}\sum(\Gamma_{ij} - \overline{\Gamma}_{ij})^2 + L_s(\Gamma, \phi) \tag{25}$$

where the L_s are bilinear forms and $L_s \longrightarrow 0$ (see Morozov [1984]).

Do we want to learn L_s? That is do we want to learn the linear condition (23) gives rise to? We do not really want to learn the value of the huge matrix corresponding to condition L_s unless we can a priori constrain the matrix (the matrix in the linear condition corresponding to (23)) to a small subspace. The only reason we needed the additional constraint (23) is that we need a large number of examples to accurately learn the value of Γ particularly if the data are noisy, which will be true if the examples are given by nature. We have to make temporary ad hoc assumptions somewhere. It is not clear where we should make them. One thing, though, is clear. We should

[30] For more complete discussions of when the Robbins-Monro procedure converges, see [Kushner and Clark, 1978; Maybeck, 1979; Herkenrath et al., 1983].

not make ad hoc assumptions of smoothness of shape.[31] We can eventually learn the right variational condition. It is safer to make the assumption in (23) because eventually $L_s \rightarrow 0$ as we accumulate more example. For a connectionist implementation of this theory, see [Aloimonos and Shulman, 1989].

3.1.7. Retrospection

Up to this point we have described ways by which we can basic·.ly learn what are the constraints that govern the problems of shape from shading, texture, patterns, etc., as well as several other problems, using some a priori knowledge. Two important issues have to be emphasized: We regularized a large part of early vision ill-posed problems (especially shape from X) without making any assumptions about explicit variational conditions; so, smoothness here is very general. A system based on this theory will learn what smoothness is.[32] We have coupled learning with discontinuous regularization, i.e. handling discontinuities that appear all the time in nature is an essential part of our learning scheme. Learning Γ means that we learn the regularized constraint $\Gamma\Xi = -\phi_{00}(\xi)$, i.e., all we have shown is how to obtain the "correct" constraint. After we have the correct constraint solving for the unknown ξ is straightforward if L is linear. If L is nonlinear, some techniques might be applicable but in general a unique solution cannot be guaranteed [Horn, 1986].

3.1.8. Empirical analysis

How can one empirically validate the proposed theory? Clearly, if one had to pick the shape from X problems, he she would have the problem of getting accurate examples and for real images this is highly nontrivial. On the other hand, synthetic image use would require a long time. Thus, we chose to use the problem of surface interpolation.

We have made some tests of our theory on the problem of surface interpolation. Here the constraint is $L = A$ (ω-observed depth). At points where there is no data $A = 0$. Where there is data, $A = 1$. More generally A might vary between 0 and 1 depending on the reliability of the observed. This would result in an equation similar to a one-dimension optical-flow equation. In our simulations, we have always assumed $A = 0$ or 1 at all points. We have ignored the positivity constraint that depth must be non-negative; implicitly we are assuming an origin a positive distance from the viewer. We generated one-dimensional piecewise polynomial curves of degree ≤ 3 (examples). The degree is chosen randomly.

Some discontinuities are discontinuities in the value, some are discontinuities in the derivatives but not in the value. Some are just discontinuities in the higher derivatives but not in the first derivative. The nature of the discontinuity and the location of discontinuities as well as the amount of the discontinuities in each derivative is determined randomly. Thus we have constructed the actual depth map. The observed map is obtained by adding noise (which may or may not be correlated) and blurring the observations. Finally we sample the points at a predefined sampling ratio p (p should be $\geq .5$ to avoid problems of sparse data). The sampling ratio is the probability that any given point will be sampled.

First we perform data independent learning and we apply the discontinuous regularization theory under the assumption $a_i = b_i = 0$ for $i > 2$; thus our variational condition ignores derivatives of L and S greater than 2. We compare the discontinuous regularization theory with the standard theory oi interpolation via regularization with S being a second derivative estimator. We have varied the parameters used to generate the simulations; in general the best discontinuous regularization is 7–30% better than the standard theory as measured by the squared difference between recovered and actual depth. We illustrate an example in Figures (15–18).

Second, we perform data dependent learning, i.e. we do not learn the best coefficients but the best function from data to coefficients. Actually we do not learn the best function but the best function in a parameterized subspace.[33] Figures (19–22) show results from this data dependent learning. In general, we get results up to 80% better than standard regularization (in the mean square error).

[31] Or any other quantity we wish to compute.

[32] Of course we are restricted to quadratic functionals for the simple reason that there does not exist any satisfactory theory for nonlinear learning to date. So we restrict ourselves to linear learning.

[33] One has to assume of course the form of the function that relates data summaries to the coefficients. In our experiments we assumed a linear form for simplicity.

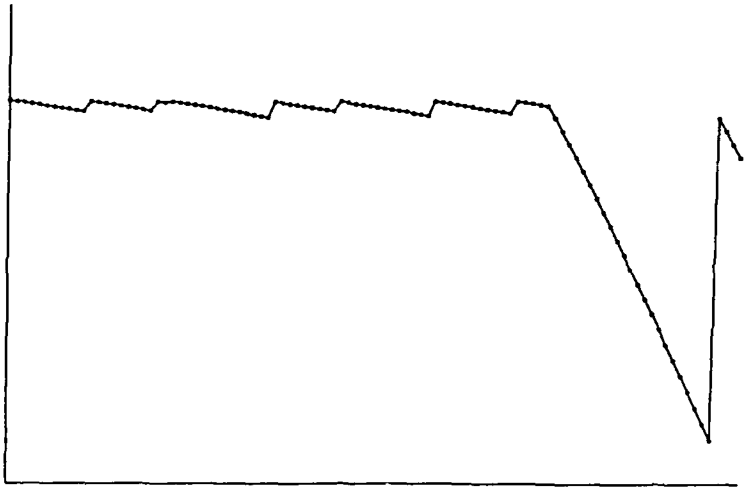


Figure 15. $x_{\min} = -4.5$, $x_{\max} = 0$; $y_{\min} = 0$, $y_{\max} = 100$.

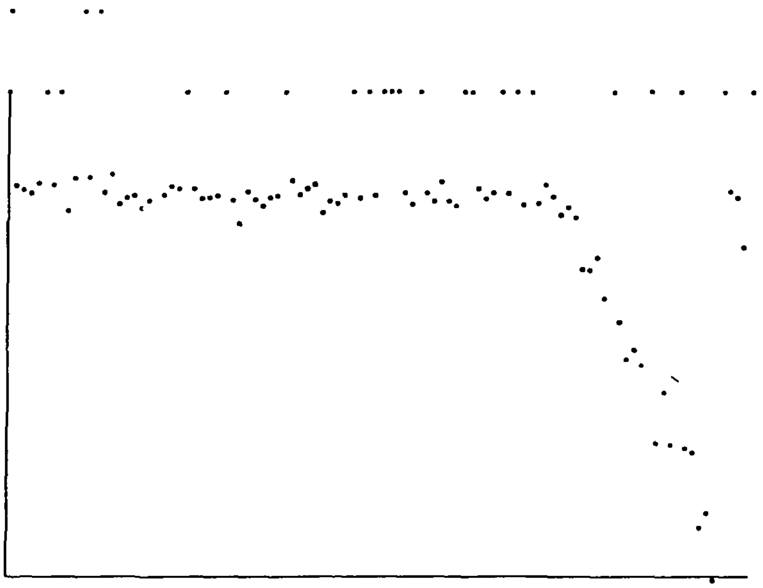


Figure 16. Observed noisified function. Points at the top row correspond to the ones not observed.

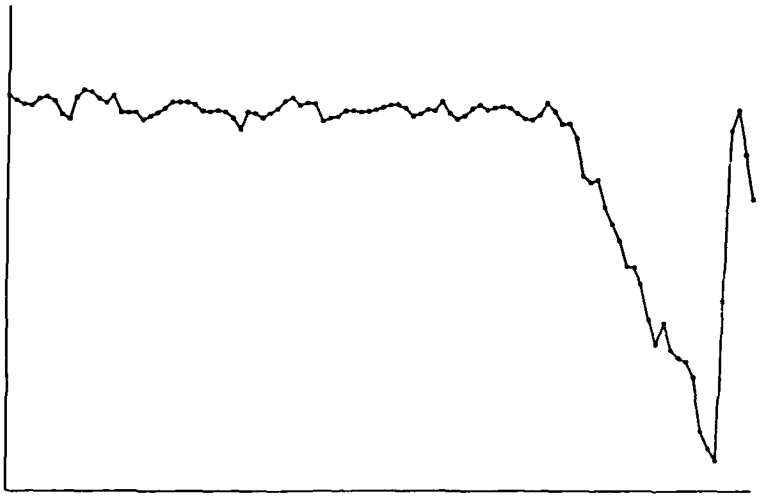


Figure 17. Recovered function based on standard regularization

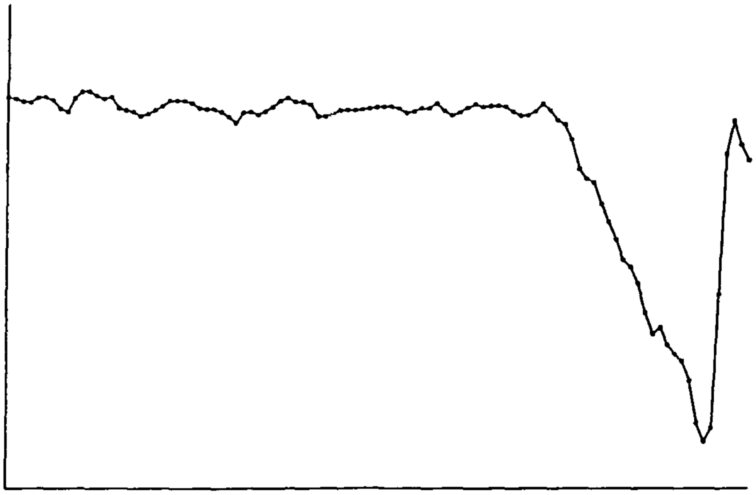


Figure 18. Recovered function based on this discontinuous regularization In this case $a_i = 0$ for $i > 0$ but $b_2 = \frac{1}{2} b_1$, $b_3 = \frac{1}{2} b_2$. The function recovered here is 17% better (in the sum of the squared error) than the function recovered by standard regularization.

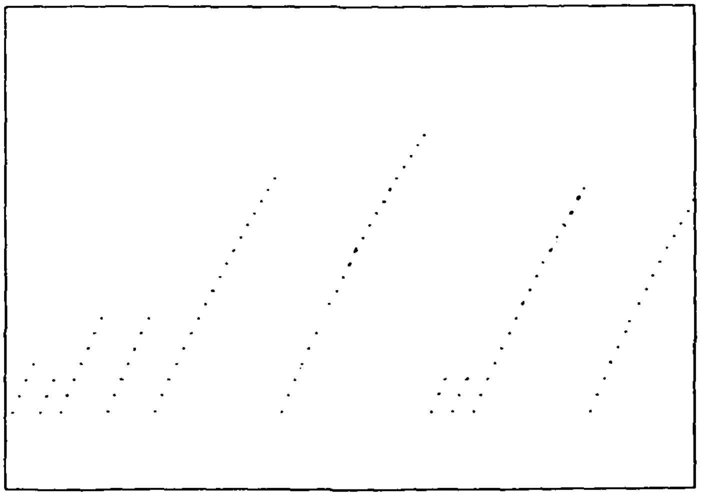


Figure 19. Original function (sampled).

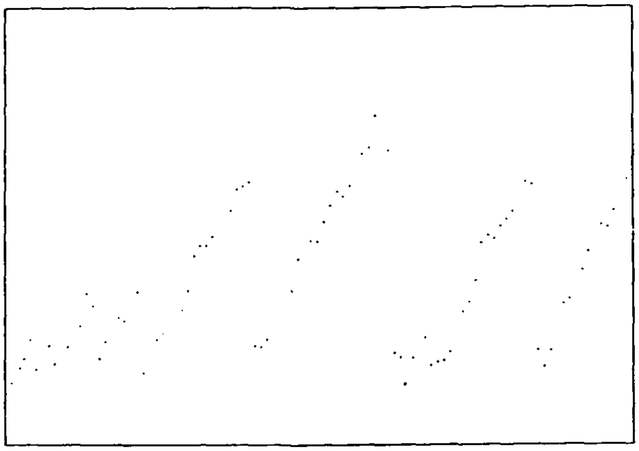


Figure 20. Function noisified

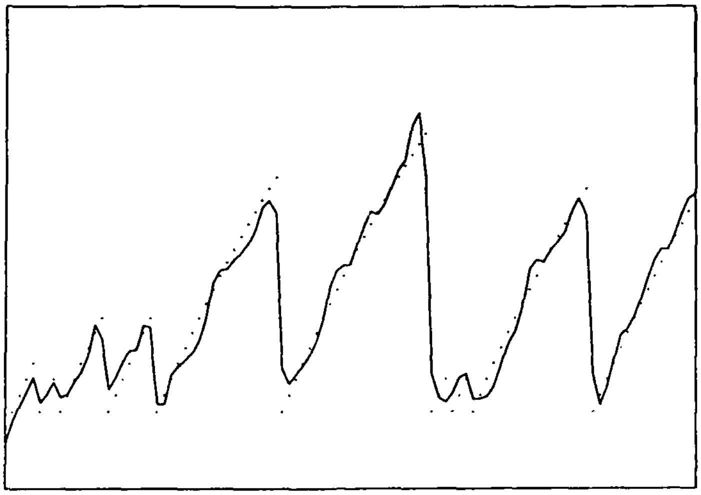


Figure 21. Recovered function based on standard regularization.

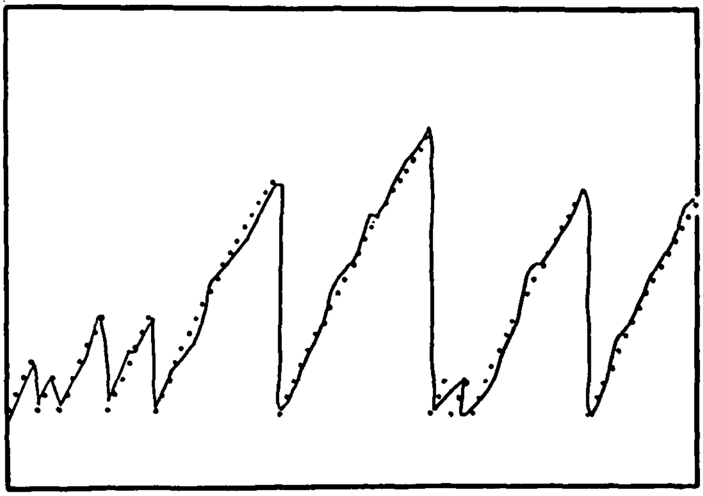


Figure 22. Recovered function based on discontinuous regularization and data dependent learning.

This discontinuous regularization theory applied to the shape from patterns problem gives results as in Figures 23, 24. Note however that the coefficients in this problem were determined in an ad-hoc way.

We are currently implementing the theory on the Connection Machine, that is we develop a general program that can be used for solving early vision problems. Our goal is to observe commonalities among the various modules, and basically learn the actual constraint. A drawback of this work is that in order for the system to learn correctly, it needs to see examples that need to span the whole space of all possible examples. As a consequence, the learning time is large. Alternatively, one might use it for industrial applications, where the example set is a restricted subspace of all possible shapes (or whatever other quantity we wish to compute).

Another goal is to use this same program for module integration. This brings us to the next section.

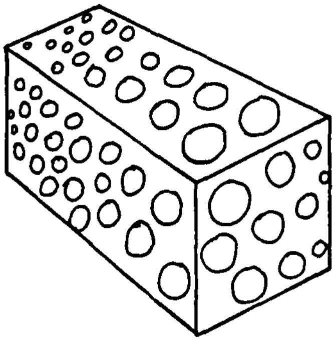


Figure 23. Image

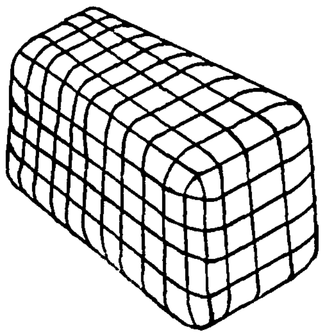


Figure 24. Reconstructed shape.

3.2. INTEGRATION

What if we want to combine shape from shading and shape from contour? Then we have two constraints relating the real world to image data: $L_1 = 0$, $L_2 = 0$ but both constraints are noisy. We need to use a priori information such as information that solutions tend to be smooth. There are three possible forms of regularization we can use: (1) Minimize $\| L_1 \|^2_{D_1} + \| L_2 \|^2_{D_2} + \| S \|^2_{D_3}$ where D_1, D_2, D_3 are norms and S is a smoothness measure such as a derivative of the real world variable ω. The simplest norms to use are proportional to the L_2 norm: $\| f \|^2_{L_2} = \int \left| f \right|^2$ by definition. So we are really solving the equation systems $L_1 = 0$, $L_2 = 0$, $S = 0$ by weighted least squares where the weights are the proportionality constants relating the D_i norms to the L_2 norm. If the unknown ω is a vector function such as optic flow and ω is the vector $(\omega_1, \omega_2, \ldots, \omega_n)$ and each component is a function of m coordinates, then if we are using a first derivative smoothness measure, we have n times m first derivatives S_{ij} that have to tend to be small; now we need to minimize $\| L_1 \|^2 + \| L_2 \|^2 +$ some quadratic function of the S_{ij}. If we use a second derivative smoothness measure we have to use a quadratic function of $m^2 n$ second derivatives $\frac{\partial^2 \omega_i}{\partial x, \partial x_k}$, etc. (2) We can minimize $\| L_1 \|^2 + \| L_2 \|^2$ subject to a limit on the size of the smoothness norm $\| S \|$. This approach is useful if we do not know the statistics of S but we do know reasonable bounds on $\| S \|$. (3) We minimize $\| S \|$ subject to a limit on the size of $\| L_1 \|^2 + \| L_2 \|^2$; thus we are finding the smoothest solution consistent with the constraints. We might apply this method if we do not know error statistics for the L_i but we can bound their norm. Methods (2) and (3) can be reduced to method (1) but there is no simple way to choose appropriate constants to do the reduction. These three regularization approaches ignore the possibility that L_1, L_2 may not be independent constraints, in which case we must define a norm on the pair of constraints L_1, L_2 rather than a norm on each constraint separately: the simplest norm will take the form $\sqrt{\int [a \, L_1^2 + b \, L_2^2 + c \, L_1 L_2]}$ where a, b, c are proportionality constants and a, $b > 0$; c depends on the correlation between the errors of the two constraints. There still remains the difficult problem of how to determine the proportionality constant, (how to determine the relative importance of smoothing and constraint satisfaction and how to determine the relative importance of and the correlation between the constraints).

We will return to the issue of how to determine the proportionality constant, but however they are determined the regularization theory described above will not work near discontinuities if we use a quadratic norm. If the proportionality constant of the smoothing term is too small, we will not smooth enough to maintain robustness in the presence of noise; if we smooth too much, we will smooth over discontinuities. The problem is the theory we have described, the usual regularization theory, as already emphasized, implicitly assumes (1) the smoothness function S is a Gaussian random variable; (2) the S's of different points are independent, identically distributed; (3) the L_i's of different points are identically distributed, independent Gaussian random variables. We have allowed for correlation between the different constraints at the same point but not for the natural tendency of errors of nearby points to be correlated. If we adopt the second or third methods of regularization, minimize inconsistency with the data (or smoothness) constraints subject to an absolute limit on violation of smoothness (or data) constraints, we will still obtain a solution that over all is reasonable adequate but either oversmooths discontinuities or undersmooths noisy data.

The basic idea behind discontinuous regularization is the same whether we are dealing with one or many sources of information. In the simplest case, we are dealing with one real-valued function ω in one dimension and we have many sources of information that provide many constraints $L_i = 0$ where L_i are functions relating data with the real world function ω. The simplest quadratic variational condition we could impose would be to minimize

$\int \left[\sum Ci\, L_i^2 + \lambda\, S^2 \right]$ where S is a smoothness measure such as the first derivative. The theory of linear discontinuous regularization instead says minimize $\int \sum_{i,j} c_{ij} \left[D^j\, L_i \right]^2 + \sum_j \lambda_j\, (D^j\, S)^2$ where the D^j operator means j^{th} derivative. The c_{ij}, λ_j are constants that need to be determined. If we allow for statistical dependency among the constraints we would instead minimize $\int \sum C_{ijkl}\, D^j\, L_i\, D^l\, L_k$ + smoothness term.

The justification for this condition is that if L and S are linear, then this variational condition leads to a linear Euler-Lagrange equation. The additional terms constrain derivatives of the constraints L_i and derivatives of the smoothnness measure S to be small. Derivatives are locally computable. If the error, L_i, of nearby points is correlated, the derivative terms allow for this correlation. But it is important to keep in mind that we are not working with arbitrary decorrelation formulae. Derivatives are quantities of genuine interest. Their peaks inform us about boundaries. We also usually need to require that the coefficients C_{ij} or C_{ijkl} be a simple function of the subscripts and this is easier to do if we use a derivative expansion of the decorrelation term. Strictly speaking, the idea here is that we first decorrelate each of the errors and then we apply regularization. Ignoring correlation between L_i's, and considering only correlations at different points of each L_i this means we need to work with the uncorrelated $\sum a_{ij}\, D^j\, L_i$ rather than the correlated L_i and thus we should minimize $\int \sum_i (\sum_j a_{ij}\, D^j\, L_i)^2$ + smoothness. If we allow for correlation among the L_i, we should minimize $\int \left[\sum_{i,j} a_{ij}\, D^j\, L_i \right]^2$ + smoothness term. Provided we choose the correct values for λ, a_{ij}, we can obtain substantially better solutions at places where error correlation is significant or where there is much information in the high frequency components of the solution. Our estimation of these high frequency components is more accurate and thus we can estimate derivatives of the solution much better and better localize discontinuities.

Even if the L_i of nearby points are not correlated, we may need to use an incorrect data consistency term (an incorrect L-term) to compensate for the fact that a quadratic S-term incorrectly overpenalizes large S. But in that case we should be very careful how we obtain the a_{ij} and λ, lest we lose accuracy in the non-discontinuous part of the solution. Care need be taken how we estimate the derivatives of the data that are used in formulating the equation. We are interested in jumps in the real world variable ω. If a constraint L_i is of the form $\omega - B$, then we can use estimates of the derivatives of B to provide clues as to the value of derivatives of ω. If these estimates are nonlinear (but local) functions of the data B, we can obtain information about the derivatives of ω which we can use to lessen the amount of smoothing over discontinuities. If the estimates are linear, we can easily incorporate information about ω and its derivatives that is more difficult to incorporate using the smoothness term. The ideal procedure turns out to be (and note this procedure is also useful when errors are correlated but correcting for an incorrect S-term is more important):

(1) We have a statistical model of the data and the unknowns and their relations. We can use this model to answer the question: what is the best local estimate of the derivative, in other words, what is the best simple estimate of the derivative at P given data at P and a small number of nearby points? Often this simple estimate will be obtained from a linear estimate followed by equating to zero all derivative estimates below a certain threshold. The best simple estimate depends on the details of the problem, however.

(2) Now we can use linear operations to improve our local non-linear estimates. At this point we might smooth our initial estimates. A purely linear derivative estimation is obtained if we regard numerical derivation as an ill-posed problem that needs to be regularized. Here the typical constraint L is of the form $\int_{X_0}^{X_1} \frac{D \text{ data}}{DX} - \left(\text{data}(X_1) - \text{data}(X_0) \right) = 0$. Poggio et al. [1984] suggests a second derivative smoothness measure S. But how do we handle $D^i\, L$ in this problem? We again need to differentiate data. We can use a simple approximation like smoothed difference quotients or use the approach suggested in the (1), (2), (3) of which this is paragraph (2).

(3) The coefficients of the terms involving derivatives of the data should depend on the reliability of these derivatives. We also take into account the reliability of the constraint $D^j\, L_i = 0$ under assumptions of perfectly accurate data derivatives when choosing coefficients.

The general formalism we have described also helps us view in a new way traditional smoothing techniques such as convolution with a Gaussian; that is the special case where we do not constrain derivatives of L ($a_{ijkl} = 0$ unless $i = j = k = l = 0$) but the smoothness term is of the form $\sum_1^N B_i\, D^i\, \omega$ where $B_i = \frac{(2\, \lambda)^i}{i!}$.

If we are working in more than one dimension, then in general we need to have more than one smoothness expression S, we have that all the $sij = \dfrac{\partial\omega_i}{\partial x_j}$ should tend to be zero where the x_j represent the different coordinates and ω_i the components of ω in the different directions. If we have more than one real world variable of interest, ω and v instead of just ω then both $\partial\omega_i / \partial X_j$ and $\partial v_i / \partial X_j$ need be small. Write $\omega_{0i} = \omega_i$, $\omega_{1_i} = vi$ etc., then the smoothness term is a quadratic expression in the $sijk = \partial\omega_{ij} / \partial X_k$ and their derivatives. As before the data consistency term involves the L_i and their derivatives. Only now there are more directions in which one can compute derivatives. If the data are very noisy or very sparse, information about derivatives of data may not suffice to adequately constrain derivatives of the real world variables of interest. There is also the possibility that constraints must be defined at coarse levels of resolution in some other cases: for example, if we are doing surface interpolation and noise removal and locally the variations in depth seem to describe a Brownian fractal (the usual assumption), but in fact unlike with a random-walk fractal, the depth seems to be trying to stay close to a certain planar surface and only make deviations from that surface that are small; so the second derivative averaged over a small region may be large, but averaged over a much larger region, it will be small. Here we would like to refer the reader to work on blue noise in halftoning [Ulichney, 1987]. Blue noise looks like white noise (standard Brownian motion case) in high frequency and looks like $\dfrac{1}{f}$ or even zero noise in low frequency. We can decorrelate the high frequency part of the spectrum.

We can define linear features η: $L(\eta) = \int \eta L$ [Amari, 1978]; a special case is where η is the Dirac-δ function, which is zero everywhere except at some point x_0 and $\int \eta = 1$. In this case, $L(\eta)$ is just L evaluated at the point x_0. We work with the linear features as if they were points. We need to define distances between these generalized points so we can take derivatives. This is easy if all the η's look the same except for scale and location of their peak: for example if the η's are Gaussians and $L(\eta)$ just represents Gaussian smoothing of L. Then $L(\eta) = 0$ means the data consistency constraint is valid after a Gaussian smoothing. There is a natural geometry for these generalized points: a generalized point η consists of an actual point x and a scale of resolution σ. The distance in the scale direction between (X_1, σ_1) and (X_2, σ_2) is $\log(\sigma_2 / \sigma_1)$. We can apply the discontinuous regularization theory just as before; only the integrals $\int a_{ij}\,(D^i\, L_j)^2$ and $\int \lambda_j\, D^j\, S$ are over the space of generalized points and the derivatives D^i, D^j refer to the derivatives in all possible directions including the scale direction. Furthermore a_{ij}, λ_j must vary with scale. So we are imposing smoothness and data consistency requirements at different levels of resolution, and requiring that the solutions vary smoothly with the level of resolution. The problems with this theory are (1) How do we determine the parameters? (2) How do we compute the derivatives of the data? (3) How do we handle the increased computational complexity because we have more complex equations than ordinary regularization? Some of these questions were answered in previous sections. For a complete analysis, see [Aloimonos and Shulman, 1989b].[34]

4. HIGH-LEVEL VISION

The preceding sections summarized a large part of our work in low- and intermediate-level vision. Our work in high-level vision is devoted to the problems of navigation and object recognition. Work on navigation was described in [Nelson and Aloimonos, 1988a,b] and [Nelson, 1988] (see also these proceedings). Here I describe some preliminary research on recognition (not model based). The analysis follows [Sullins, 1989], where the ORACLE[35] system is developed. The basic idea is that one tries to learn the appearance of an object (2D) by first representing it with a Boolean formula. Then the Boolean formula is learned from examples.

The goal of the algorithm is to learn the Boolean expression that corresponds to a given input-output behavior, in particular a Boolean expression in *disjunctive normal form*. It is designed to run on a distributed system, having intermediate processors whose inputs correspond to the "and" part of the DNF expression and whose outputs will correspond to the "or" part. Each processor will be programmed to detect exactly one of the *conjunctive terms* of the DNF formula. For example, consider the DNF Boolean expression (A and $\overline{B}$) *or* ($\overline{C}$ and D). Its conjunctive terms are (A, $\overline{B}$) and ($\overline{C}$, D). The expression is true if the input corresponding to A is active and the input corresponding to B is inactive (the states of the inputs corresponding to C and D don't matter) or if the input corresponding to C is inactive and the input corresponding to D is active (the states of A and B don't

[34] Our theory of discontinuous regularization is being applied to various problems by several research groups (see for example [Yuille and Gryswatz, 1988] for an application to motion coherence.

[35] Organized Adaptive Constraint Learning

matter). We generally represent conjunctive terms in vector form, with inputs that don't matter replaced by a "–". Assuming that the inputs correspond to the variables in alphabetical order, the vector form of $(A, \bar{B})$ and $(\bar{C}, D)$ would be (1, 0, –, –) and (–, –, 0, 1) (see Figure 25). In a corresponding neural network there would be one processor set up to detect $(A, \bar{B})$ and another to detect $(\bar{C}, D)$. If either of these were found by a processor, that processor would activate the output.

4.1. CHOOSING A SEED VECTOR FOR THE SYSTEM

The simplest way for a processor to choose a seed vector is to simply wait until it is presented with an input vector that receives positive feedback indicating that the output should be active, and use the state of the inputs at that time. For example, the possible seed vectors of (A and B) *or* (C and D) are:

(1, 1, 0, 0)
(1, 1, 0, 1)
(1, 1, 1, 0)
(1, 1, 1, 1)

(0, 0, 1, 1)
(0, 1, 1, 1)
(1, 0, 1, 1)

Having more than one processor with the same conjunctive term would not be an efficient allocation of resources. Accordingly, we will want to avoid choosing vectors for which the system already activates the correct outputs, as we can assume that those cases are already covered by conjunctive terms detected by other processors. In addition, we will probably want to add a bit of nondeterminism to the selection of vectors in order to keep all of the free processors from simultaneously choosing the first vector available. When a vector receives positive feedback, it will have a certain probability of being chosen as a seed vector. If there is a large degree of feedback error in the environment, then a processor could choose a "negative" vector instead of a "positive" one.

4.2. LEARNING CONSTRAINTS

4.2.1. Constraints

Once a seed vector is chosen, we must determine which inputs are essential to the positive feedback and which are not. To be clearer, we may consider each member of the seed vector to be a ***constraint***. We already know that when all of the constraints are met—that is, when each of the inputs are in the state that they are in the seed vector—the output should be activated. From this we could set the processor to detect that particular seed vector and activate the appropriate outputs when it is found.

Of course, to devote an entire processor to a single input vector would be extremely wasteful. We want to expand the set of input vectors that the processor detects by eliminating as many constraints as possible. As eliminating a constraint allows the processor to accept vectors with that input in both the state of the constraint and in the opposite state, it effectively doubles the number of vectors that the processor can accept. A processor with k constraints deleted can accept up to 2^k input vectors. This kind of exponential increase may allow a polynomial number of processors to cover an exponential number of possible input vectors.

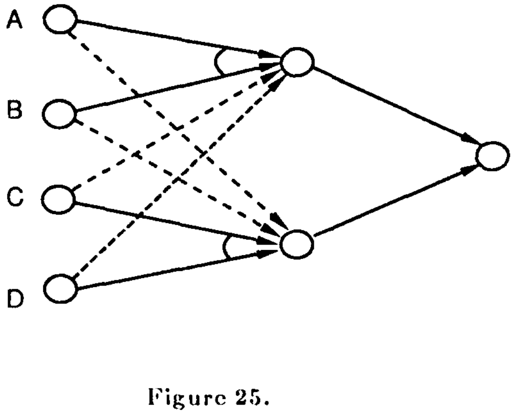


Figure 25.

4.2.2. Motion in constraint space

The core of the ORACLE system is the addition and subtraction of constraints in order to minimize the difference between the expected and the actual input-output behaviors. In this section we describe the criteria used to change the constraints, that is, the *constraint motion*. For each possible constraint (i.e. for the state of each input in the processor's seed vector) there are four different types of constraint motion data. At any point in time, some of those inputs will be enforced constraints while the rest will be potential constraints not enforced. Adding a constraint at an input means that the processor will no longer accept input vectors that have a different value from the seed vector at that input. Conversely, removing a constraint means that the processor will now accept input vectors that differ from the seed vector at the input. If the constraint is *not* currently enforced, the environment may present data indicating that the constraint should be added and data indicating that it should *not* be added. If the constraint currently *is* enforced, there is data indicating that it should be removed and data indicating that it should not be removed. Simply put, we will add constraints that cause the network to reject more negative vectors than positive vectors and we will remove constraints that cause the network to reject more positive vectors than negative vectors.

4.2.2.1. Adding constraints. Generally speaking, constraints should be added to processors that accept too many input vectors receiving negative feedback. Because of the DNF structure of the network, a vector incorrectly accepted by any processor is also incorrectly accepted by the entire network. A negative vector must differ from a processor's seed vector in the value of one or more inputs (otherwise, that seed vector would not have received positive feedback). Each of those inputs is a potential constraint that would prevent the negative vector from being accepted in the future, so each of these potential constraints receives a *positive vote* for change at that processor.

On the other hand, we do not wish to add a constraint to a processor if the constraint would prevent too many positive vectors from being accepted, specifically those positive vectors *not accepted by any other processor*. If that processor were no longer able to accept such "uniquely accepted" input vectors, then the entire network would incorrectly reject them as well. For all positive input vectors uniquely accepted by the processor, each input that has a different value from that of the seed vector (and thus would cause the positive vector to be rejected if it were to become a constraint) receives a *negative vote* for change. Positive and negative votes are collected for each potential constraint over a sampling of the input vectors. At the end of that time, potential constraints with significantly more positive than negative votes are added to the processor.

4.2.2.2. Removing constraints. One might expect that the removal of constraints would be somewhat symmetrical to the addition of them, removing constraints that prevent positive vectors from being accepted and not removing those that prevent negative vectors from being accepted. The conjunctive structure of the processors forces us to modify this idea, however. While a negative vector can be prevented by adding a single constraint to the processor that the vector fails to meet, removing a single constraint might not enable a processor to accept a particular positive vector. A positive vector might not meet *many* existing constraints of a processor. We want to remove constraints when doing so would cause more positive vectors to be accepted, but for most positive vectors removing a single constraint will not make any difference. For the (A and B) *or* (C and D) example, if A, B, C, and D are all constraints then removing either C or D will not cause (A and B) to be accepted. *Both* of them must be removed. The only time that removing a single constraint would allow a positive vector to be accepted is if the vector met all but one of the constraints. If there are many constraints (some of which might not be correct), such "near misses" [Winston, 1984] could be very rare. Keeping track of whether removing combinations of two or more constraints would improve performance is not an option, either, due to the exponential number of statistics that would have to be kept.

The system needs some criteria for voting to remove a constraint, even if doing so would not cause any more positive vectors to be accepted. The logical solution is to allow all positive vectors that are not accepted by the network to influence the removal of constraints, but giving "near miss" vectors more influence than others. If a positive vector is not accepted by the network, then each constraint at processor i that prevented it from being accepted at that processor is given a positive vote for removal proportional to σ^{-n_i}, where n is the number of constraints that the vector failed to meet at processor i. σ is a measure of how relevant "near misses" are compared to other misses. For large σ near misses are the only ones that produce votes. That is, each processor is changed in proportion to how close it already is to accepting to vector; this helps the system to properly distribute the responsibility for accepting vectors by assuring that only a few processors are forced to learn each one.

Negative votes against removing constraints are collected in a similar manner. If a constraint helps to prevent a negative vector from being accepted, it receives a negative vote against removal proportional to σ^{n_i}. The fewer constraints preventing a negative vector from being accepted, the more difficult they should be to remove. As in the case of adding constraints, the input is sampled over a certain period of time. If the number of positive votes is greater than the number of negative votes, the constraint is removed.

4.3. SUMMARY

To summarize the four types of constraint motion:

(1) If the processor accepts a negative vector, then each inactive constraint that the vector fails to meet is given a positive vote for addition.

(2) If the processor is the only one to accept a positive vector, then each inactive constraint that the vector fails to meet is given a negative vote against addition.

(3) If the processor rejects a positive vector that no other processor accepts, then each active constraint that the vector fails to meet is given a positive vote for removal, and the strength of that vote is an exponential function of the negative of the number of constraints the vector fails to meet.

(4) If the processor rejects a negative vector, then each active constraint that the vector fails to meet is given a negative vote against removal, and the strength of that vote is an exponential function of the negative of the number of constraints the vector fails to meet.

The data is collected for a large sample of the input vectors. This insures that incorrect feedback will have little effect on the system, as the incorrect data will usually be outvoted by the correct data. Each processor starts with no constraints and adds or removes them in such a manner that it eventually learns one of the conjunctive terms that logically implied its seed vector. Basing the constraints of the processors on the values of their individual seed vectors helps them to avoid the crosstalk problems of gradient descent algorithms. Each processor learns from those input vectors directly relevant to its seed vector, ignoring those that can be more easily accepted by another.

In [Sullins, 1989] a probabilistic analysis of the performance of the above algorithm is performed and it is shown, under some assumptions, that the system performs very satisfactorily when the problem at hand can be expressed in DNF Boolean formula with not an extremely large number of conjunctive terms.

The primary problem used to test ORACLE involved determining whether or not a given 6 by 6 binary pixel image (that is, 36 inputs) contained active input units in the shape of a square. There were 14 possible squares (9 of size 4, 4 of size 5, 1 of size 6), and an input vector was given a 50% chance of being assigned one of these squares. Vectors containing squares were given "background noise" of 25%, that is, each non-square input was active with a probability of 25%. This means that there were on the order of 2^{24} (over 16 million) possible input vectors containing squares. The inputs of vectors not assigned squares were active with a probability of 50%, meaning that there were 2^{36} (over 60 billion) of those.

The ideal set of constraints for a processor would clearly be those and only those inputs corresponding to parts of the square in the processor's seed vector. Since there are 14 different squares, the disjunctive normal form of this behavior would contain 14 conjunctive terms. These input vectors were presented to a network containing 40 processors, which according to a theoretical analysis should have been a sufficient number to cover the conjunctive terms with a minimum of reassignment.

The learning curves for various levels of feedback error are given in Figure 26. The X-axis represents the number of examples shown. The Y-axis represents the percentage of time that the network gave the correct response to the question of whether or not the input contained a square. Note that this is not the percentage of time that the output matched the (possibly incorrect) feedback; we are interested in how well the system managed to ignore incorrect feedback, not how well it duplicated it. Each curve is labeled with its level of feedback error, the percentage of time that the feedback was incorrect.

As the figure shows, ORACLE learns to detect squares after seeing only a tiny fr [illegible] . the possible input vectors. In fact, the correctness of the network is generally close to 100%, or at least much greater than the correctness of the feedback. This indicates that the system succeeds in choosing the correct constraint motion in the long run despite occasional errors. While the learning time increases with the feedback error, the behavior is still learned quickly for even large amounts of error. The learning does not begin to deteriorate until the feedback error is greater than 30%.

The individual processors behaved as predicted, quickly acquiring a set of correct constraints that distinguished it from the others and then going through the slow process of removing the incorrect ones it picked up along the way. This is reflected in the learning curves, which quickly reach a good level of correctness, and then slowly continue to improve to higher levels. They do not quite reach 100% correctness because incorrect constraints are still added from time to time.

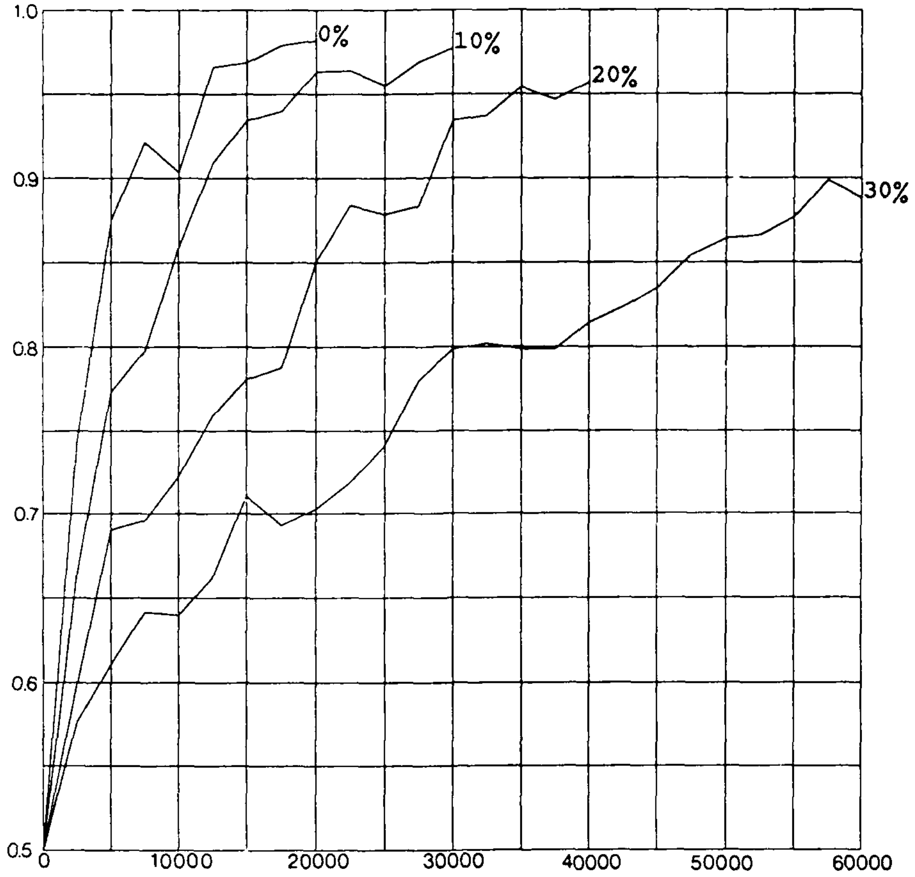


Figure 26.

The next experiment involved reducing the number of processors to 5, far below the minimum needed to cover all of the 14 conjunctive terms. The purpose of this was to force the network to learn *features* of squares. With a feedback error of 10%, the constraint sets in Figure 27 were created after 20,000 examples. As can be seen, ORACLE has discovered the ideas of *corners* and *parallel lines*. Both types of templates allow a processor to accept more than one kind of square. Generalizing to these features did not increase the error significantly, as it was unlikely that the features would be generated at random (2^{-7} for the corners, which is less than the 10% feedback error).

ORACLE was also given the more ambiguous problem of detecting fish tails. A set of 26 pictures ([Cousteau, 1953], [Cousteau, 1963]) of fish tails were translated (by hand) to the 6 by 6 binary format. These included many different species with greatly dissimilar tails, in order to insure that more than one type of detector was needed. Background noise was added by activating inputs with probability 0.25, giving 2^{36} possible fish tails. The network was either presented with one of these with probability .5 (it was presented with random noise otherwise). Fish tails were given positive feedback and noise was given negative feedback, except for a feedback error of 10%. The network contained 10 processors.

-	-	-	-	-	-
-	-	-	-	-	-
1	-	-	-	-	-
1	-	-	-	-	-
1	-	-	-	-	-
1	1	1	1	-	-

-	1	1	1	1	-
-	-	-	-	1	-
-	-	-	-	1	-
-	-	-	-	1	-
-	-	-	-	-	-
-	-	-	-	-	-

-	-	-	-	-	-
-	1	-	-	-	1
-	1	-	-	-	1
-	1	-	-	-	1
-	1	-	-	-	1
-	-	-	-	-	-

Figure 27.

The problem was made more interesting by also deactivating any input with probability 0.05. This means that there were no good conjunctive terms for the network to form, as there would always be fish tails that violated any set of constraints at the deactivated inputs. Because we allow any of the inputs to be corrupted, ORACLE was forced to find the most basic prototypes of fish tails. These were far less well-defined than the features or conjunctive terms of the squares. Some of them are given in Figure 28. The learning curve is shown in Figure 29.

As can be seen from Figure 28, ORACLE finds widely varied general features of fish tails. During each run, 3 or 4 processors would acquire one of these basic features, but none of them accounted for a majority of the positive vectors accepted. This shows that the system was able to properly distribute the detection of different types of fish tails over different processors.

-	-	-	-	-	-
1	-	-	1	-	-
1	1	1	1	-	-
1	-	-	1	-	-
-	-	-	-	-	-
-	-	-	-	-	-

-	-	0	-	-	1
-	-	-	-	1	-
1	1	1	1	-	-
-	-	-	1	-	-
-	-	-	-	-	-
-	-	-	-	-	0

-	-	-	-	1	1
-	-	1	-	-	-
1	1	1	-	-	-
1	1	-	-	-	-
-	-	-	-	-	-
-	-	-	-	-	-

-	-	-	-	-	1
1	-	-	-	-	-
1	1	1	-	-	-
1	-	-	1	1	-
-	-	-	-	-	-
-	-	0	-	-	-

-	-	-	-	-	-
-	-	-	1	-	1
-	1	-	-	-	1
1	1	-	-	1	1
-	-	-	-	-	1
-	-	-	-	-	-

Figure 28.

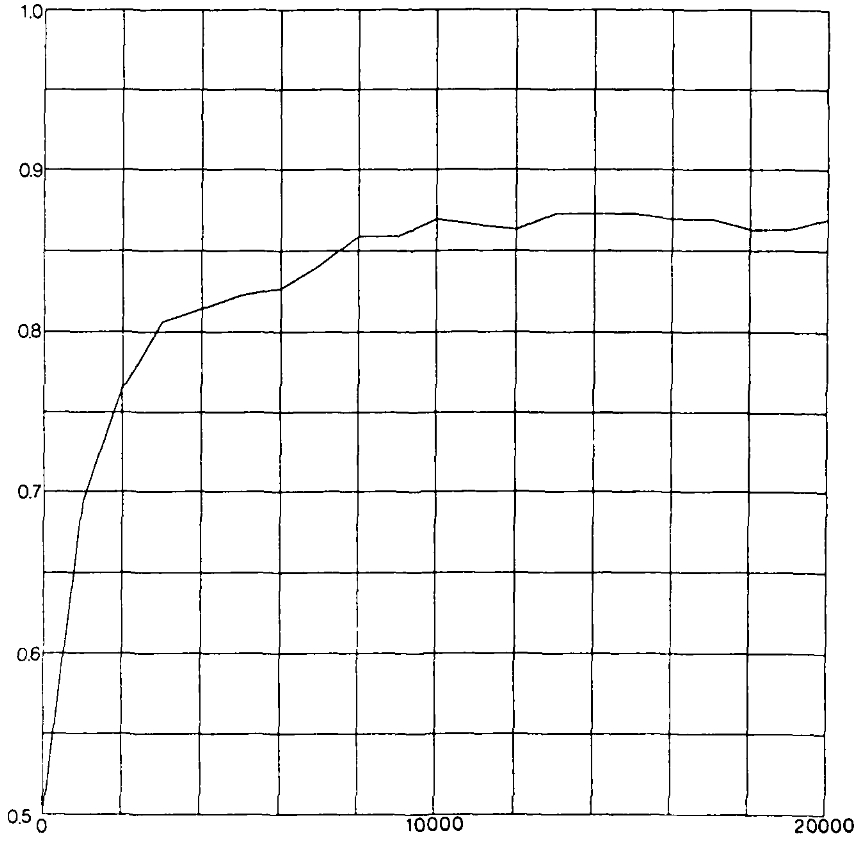


Figure 29.

5. CONCLUSIONS

I described here a part of our research in low-, intermediate-, and high-level vision. The theory of discontinuous regularization serves as a unifying theme for describing computational theories behind early vision processes as well as for integrating the low-level modules. I also showed, motivated by T. Poggio,[36] how one can synthesize operators from examples and basically learn the actual constraints that govern inverse visual problems. We are currently implementing on the Connection Machine the DRM (discontinuous regularization machine) whose purpose would be to extract object descriptions for the purposes of recognition. Our recognition work could be characterized as "learning" how to recognize an object from examples. Having laid down the foundations for 2-D recognition we are moving to the direction of recognizing 3-D objects.

Our work on navigation is currently focused on sensory feedback path planning and the design of approximate algorithms for addressing the intractable problems associated with kineodynamic path planning.[37]

Acknowledgements

Thanks are due to the following members of the Robot Vision group at Maryland for their contributions, endless discussions and for constructive criticism: A. Rosenfeld, M. Spetsakis, D. Shulman, A. Basu, J.-Y. Hervé and J. Sullins.

References

A. Albert, *Regression and the Moore-Penrose Inverse*, Academic Press 1972.

J. Aloimonos, "Computing intrinsic images", Ph.D. thesis, Dept. of Computer Science, University of Rochester, Rochester, NY, 1986.

J. Aloimonos, "Visual shape computation", *Proc. IEEE*, Vol. 76, No. 8, August 1988.

J. Aloimonos and A. Basu, "Combining information in low-level vision", Technical Report CAR-TR-336, Computer Vision Laboratory, Center for Automation Research, University of Maryland, College Park, 1988.

J. Aloimonos and D. Shulman, "Learning early vision computations", *J. Opt. Soc. America, A*, in press.

S. Amari, "Feature spaces which admit and detect invariant signal transformations", 4th Int'l. Conf. on Pattern Recognition, Tokyo, 452–456, 1978.

A. Ben Israel and T. Greville, *Generalized Inverse Theory and Applications*, Wiley, 1978.

A. Blake and A. Zimmerman, *Visual Reconstruction*, M.I.T., 1987.

O. Bottema and B. Roth, *Theoretical Kinematics*, North Holland, New York, 1979.

A.W. Bowman, P. Hall, D.M. Titteington, "Cross-validation in non-parametric estimation of probabilities and probability densities", *Biometrika* **71**, 341–351, 1981.

M. Brady, "Computational approaches to image understanding", *ACM Computing Surveys* **14**, March 1982.

A. Brandt, D. Ron and D. Amit, "Multilevel approaches to discrete-state and stochastic problems", *Proc. 2nd European Conference on Multigrid Methods*, Cologne, October 1–4, 1985, W. Hackbusch and U. Trottenberg, eds., *Lecture Notes in Mathematics*, Springer-Verlag, 1986.

A. Brandt, "Multilevel adaptive methods for boundary problems", *Math. Comput.*, **31**, 333–390, 1977.

L. Bromberg, *Bayesian analysis of linear models*, Marcel Dekker, 1985.

J. Y. Cousteau, *The Silent World*, Harper and Row, New York, 1953.

J. Y. Cousteau, *The Living Sea*, Harper and Row, New York, 1963.

J.A. Feldman, "Four frames suffice: A provisional model of vision and space", *Behavioral and Brain Sciences*, June 1985.

W. Feller, *An Introduction to Probability Theory and Its Applications* (2 vols.), Wiley, 1966.

S. Geman and D. Geman, "Stochastic relaxation, Gibbs distribution and Bayesian restoration of images," *IEEE PAMI* **6**, 1984.

[36] See (Hurlbert and Poggio, 1988).

[37] Basu, A. and Aloimonos, J., "Approximate constrained motion planning", and Basu, A., Sharma, R. and Aloimonos, J., "On the warehouseman's problem: some easy cases", Technical Reports, Computer Vision Laboratory, Center for Automation Research, University of Maryland, College Park, in press.

J.J. Gibson, *The Perception of the Visual World*, Houghton-Mifflin, Boston, MA, 1950.

J. Hadamard, *Lectures on the Cauchy Problem in Linear Partial Differential Equations*, Yale, 1923.

P. Hall, "Large scale optimality of least squares cross-validation in density optimization", *Ann. Stat.* **11**, 1156–1174, 1983.

R.M. Haralick and S.J. Lee, "The facet approach to optical flow", in L.S. Baumann (ed.) *Proc. DARPA Image Understanding Workshop*, 1983.

U. Herkenrath, D. Kalin and W. Vogel (Eds.), *Mathematical Learning Models: Theory and Applications*, Springer 1983.

P. Huber, *Robust Statistics*, Wiley, 1981.

A. Hurlbert and T. Poggio, "Learning a color algorithm from examples", *Science*, to appear, 1988.

R.E. Kalman, "A new approach to linear filters and prediction problems", in *J. Basic Eng. Trans. ASME*, Series D, 8235–45.

S. Kirkpatrick, "Optimization by simulated annealing: Quantitative studies", *J. Statis. Phys.* 34, 975–986, 1984.

T. Kohonen, *Associative Memory: A Systems Theoretical Approach*, Springer, 1977.

H. Kushner and D. Clark, *Stochastic Approximation Methods for Constrained and Unconstrained Systems*, Springer, 1978.

E.H. Land and McCann, J.J. "Lightness and retinex theory", *J. Opt. Soc. Am.* **61**, 1–11, 1971.

D. Lee and T. Pavlidis, "Smoothing splines with discontinuities for image analysis," *Proc. IEEE ICCV*, 1987.

L. Ljung, "Analysis of recursive stochastic algorithms", *IEEE Trans. Aut. Control* **AC-22**, 551–575, 1977.

D. Marr, *Vision*, W.H. Freeman, San Francisco, 1982.

J. Marroquin, "Surface reconstruction preserving discontinuities", AI Memo 792, M.I.T. Artificial Intelligence Laboratory, 1984.

J. Marroquin, "Probabilistic solution of inverse problems", MIT Ph.D. Thesis, 1985.

J.C. Marshall, and F. Newcombe, "Patterns of paralexia", *Psycholinguis. Res.* **2**, 175–199, 1973.

L. Matthies, R. Szelisky and T. Kanade, "Incremental estimation of dense depth maps from image sequences", *Proc. IEEE CVPR*, 1988.

P. Maybeck, *Stochastic Models, Estimation and Control*, Academic Press, 1979.

N. Megiddo and A. Tamir, "On the complexity of locating linear facilities in the plane", *Operation Research Letters*, **1**, 1982, 194–197.

D. Mumford and J. Shah, "Boundary detection by minimizing functionals", *Proc. IEEE CVPR*, 1985, 22–25.

H.H. Nagel and W. Enkelmann, "An investigation of smoothness constraints for the estimation of displacement vector fields from image sequences", *IEEE PAMI* **8**, 1986, 565–593.

R. Nelson, "Visual homing using an associative memory", Technical Report CAR-TR-380, Computer Vision Laboratory, Center for Automation Research, University of Maryland, College Park, 1988.

R. Nelson and J. Aloimonos, "Finding motion parameters from spherical flow fields (Or the advantages of having eyes in the back of your head)", Technical Report CAR-TR-287, Computer Vision Laboratory, Center for Automation Research, University of Maryland, College Park, 1988a.

R.C. Nelson and J.Y. Aloimonos, "Using flow field divergence for obstacle avoidance in visual navigation", Technical Report CAR-TR-322, Computer Vision Laboratory, Center for Automation Research, University of Maryland, College Park, 1988b.

T. Poggio and V. Torre, "Ill-posed problems and variational principles in vision", AI Memo 773, M.I.T. Artificial Intelligence Laboratory, 1984.

T. Poggio, V. Torre and C. Koch, "Computational vision and regularization theory", *Nature* **317**, 314–319, 1986.

B.T. Polyak and Y.Z. Tsypkin, "Optimal pseudogradient algorithms", *Automat. Remote Control* **8**, 74–84, 1980.

B.T. Polyak and Y.Z. Tsypkin, "Adaptive estimation algorithms (convergence, optimality, stability)", *Automat. Remote Control* **3**, 71–84, 1979.

K. Prazdny, "Determining the instantaneous direction of motion from optical flow generated by a curvilinearly moving observer", *CVGIP* **17**, 1981, 94–97.

C.R. Rao, *Linear Statistical Theory and Its Applications*, Wiley, 1973.

R. Rousseeauw, *Robust Regression and Outlier Detection*, Wiley, 1987.

H. Shariat, Ph.D. Thesis, University of Southern California, 1986.

D. Shulman and J. Aloimonos, "Boundary preserving regularization: Theory, part I" Technical Report CAR-TR-356, Computer Vision Laboratory, Center for Automation Research, University of Maryland, College Park, 1988.

D. Shulman and J.-Y. Hervé, "Regularization of discontinuous flow fields", *Proc. IEEE Workshop on Motion*, Irvine, CA, March 1989.

B.W. Silverman, *Density Estimation for Statistics and Data Analysis*, Chapman, 1986.

M. E. Spetsakis and J. Aloimonos, "Closed form solution to the structure from motion problem from line correspondences", *Proc. AAAI 87*, Seattle, WA, 1987.

M. Spetsakis and J. Aloimonos, "Optimal estimation of structure from motion from point correspondences in two frames", *Proc. IEEE ICCV*, Tampa, FL, 1988a.

M. Spetsakis and J. Aloimonos, "A multi-frame approach to visual motion perception", *Proc. IEEE Workshop on Motion*, Irvine, CA, March, 1989.

K.A. Stevens, "The visual interpretation of surface contours", *Artif. Intell.* **17**, 47–75, 1981.

D. Terzopoulos, "Regularization of inverse problems involving discontinuities", *IEEE PAMI* **8**, 1986, 413–424.

A.N. Tikhonov and V.Y Arsenin. *Solution of Ill-Posed Problems*, Winston, 1977.

M.L. Tiku, *Robust Inference*, Dekker, 1986.

R.Y. Tsai and T.S. Huang, "Uniqueness and estimation of three dimensional motion parameters of rigid objects", in *Image Understanding 1984*, S. Ullman and W. Richards (eds.), Ablex Publishing Co., New Jersey, 1984.

Y.Z. Tsypkin, *Foundation of the Theory of Learning Systems*, Academic Press, 1973.

R. Ulichney, *Digital Halftoning*, MIT, 1987.

E.J. van Asselt, "The multigrid method and artificial viscosity", in W. Hackbusch and U. Trottenberg (eds.) *Proc. Conf. on Multigrid Methods 1981*, Springer, 1982.

G. Wahba, "*Cross-validation spline methods for the estimation of functions from data on functionals*", in H. David and D. David (eds.), *Statistics: An Appraisal*, Iowa, 1984.

M.T. Wasan, *Stochastic Approximation*, Cambridge, 1969.

L. Weiskrantz, E.K. Warrington, M.D. Sanders, and J. Marshall, "Visual capacity in the hemianopic field following a restricted occipital ablation", *Brain* **97**, 709–728, 1974.

S.R. White, "Concepts of scale in simulated annealing", *Proc. IEEE Int'l. Conf. on Computer Design*, Port Chester, 1984.

P. H. Winston, *Artificial Intelligence*, 2nd edition, Addison-Wesley, Reading, MA, 1984.

M. Yachida, "Determining velocity in spatio-temporal neighborhoods from image sequences", *CVGIP* **21**, 1983, 262–279.

G.S. Young and R. Chellappa, "3-D motion estimation using a sequence of noisy stereo images", *Proc. IEEE CVPR*, 1988.

A. Yuille and N. Gryswatz, "The motion coherence theory", *Proc.* 2^{nd} *IEEE ICCV*, Tarpon Springs, FL, 1988.

COLOR IMAGE SEGMENTATION USING MARKOV RANDOM FIELDS*

Michael J. Daily

Hughes Research Laboratories
3011 Malibu Canyon Road
Malibu, CA. 90265

Abstract: We discuss the use of Markov Random Fields (MRFs) in color image segmentation of natural, outdoor scenes. MRFs provide an elegant means of specifying a local energy function which embodies the expected dependencies of neighboring pixels and includes both the prior and posterior probabilistic distributions. This local, neighborhood-based specification of dependencies avoids *ad hoc*, brittle methods using global image knowledge. We present a brief analysis of ongoing research in color differencing methods since they are central to the problem of color segmentation. We develop and compare the use of three different lattice structures for coupled MRFs with line and color processes based on squares, hexagons, and triangles and also discuss current efforts in MRF parameter derstanding.

1. INTRODUCTION

The use of color information can significantly improve discrimination and recognition capability over purely intensity-based methods. Methods for low-level segmentation of color imagery are numerous. Techniques using recursive region splitting with histogram analysis [18],[19] and edge and boundary formation [5] have been applied to natural color imagery with some success. These methods typically suffer from a lack of important spatial knowledge in histograms and over-dependence on global thresholds and image assumptions. Statistical methods, such as classical Bayes decision theory, which are based on previous observation have also been quite popular [2],[21]. However, these methods depend on global *a priori* knowledge about the image content and organization. Until recently, very little work had used underlying physical models of the color image formation process in developing color difference metrics. Physically-based algorithms have produced excellent segmentations for color imagery obtained under controlled conditions [7],[14],[15]. In this paper, we discuss the use of Markov Random Fields (MRFs) in color image segmentation of natural, outdoor scenes. MRFs provide an elegant means of specifying a local energy function which embodies the expected dependencies of neighboring pixels and includes both the prior and posterior probabilistic distributions. This local, neighborhood-based specification of dependencies avoids *ad hoc*, brittle methods using global image knowledge. We present a brief analysis of ongoing research in color differencing methods since they are central to the problem of color segmentation. We develop and compare the use of three different lattice structures for MRFs based on squares, hexagons, and triangles and discuss current efforts in MRF parameter understanding.

2. BACKGROUND FOR MARKOV RANDOM FIELDS

Markov Random Fields possess several characteristics which make them useful in color image segmentation. Properties such as smoothness and continuity of color regions over an entire image can be enforced using only dependencies among local neighbors. Discontinuities which separate regions of constant color may be computed while smooth regions are being found. In addition, the inclusion of both the prior and posterior distributions (through Bayes' rule) establishes a relationship between noisy observed imagery and the color segmentation results.

* This research was supported in part by the Defense Advanced Research Projects Agency under contract DACA76-85-C-007.

Details of MRF theory can be found in [4] and [17]. Briefly, a Markov Random Field is a lattice of sites; for example, an image of pixels. Since MRFs are stochastic processes, the pixels in an image may take on any of their allowed values, which means that all images can be generated. In addition, the conditional probability of a particular pixel having a certain value is only a function of the neighboring pixels, not of the entire image. The Hammersley-Clifford theorem establishes the equivalence between the conditional probabilities of the local characteristics in the MRF and local energy potentials in a Gibbs distribution. Therefore, the *a priori* probability that the MRF is in a particular state can be calculated by summing the local energies over the entire image. We are interested in obtaining the MRF state that maximizes the *a posteriori* probability of the final segmentation given the observed data. From additional theorems, the *a priori* energies can be added to an *a posteriori* energy term involving the difference between the actual observed data and the current MRF state (or predicted image). Therefore, the Gibbs distribution energy function consists of two parts, one describing the interaction potential between neighbors, and the other associated with the difference between the predicted image and the actual observed data. Several methods of minimizing the energy function over the image (i.e. maximizing the probability) can be used, among them simulated annealing. deterministic procedures, and network solutions.

MRFs have been used for a variety of vision tasks from image and surface reconstruction to fusion of multiple low-level vision modules. Geman and Geman [4] have used the MRF approach on simple synthetic intensity images for image reconstruction. They added a useful twist to the standard approach by coupling two MRFs, one for the intensity process, and one for a binary line process. The binary line process marks the location of discontinuities in the intensity surface, making the energy non-convex and highly non-linear. Using a method they call the Gibbs Sampler, they perform the non-convex energy minimization using a Metropolis-like, simulated annealing algorithm. In the theoretical case, they prove the Gibbs Sampler will produce an annealing schedule which guarantees the global minimum; however, due to practical limitations, the schedule is sub-optimal. Marroquin [17] was the first to apply coupled MRFs with line processes to the problem of surface reconstruction from sparse depth data. More recently, Gamble and Poggio [3],[20] have used coupled MRFs for fusing low-level visual information. The formation of line process discontinuities and smoothing of processes in depth and motion data, color, and texture are coupled through separate lattices of MRFs to the intensity edges, which guide their formation. Following Marroquin, they chose sub-optimal deterministic procedures to minimize the energy functionals. Another method for surface reconstruction using sparse synthetic depth and intensity data is due to Chou and Brown [1]. Using MRFs and a technique called Highest Confidence First (HCF) to minimize the energy, they chose to update sites with the least stability (highest confidence for changing states) with respect to the current state before updating sites which were more stable. Koch, Marroquin, and Yuille [16] have used the coupled MRF approach to perform surface reconstruction from sparse depth data as well. However, they minimize the energy using linear graded neurons, as in the work of Hopfield and Tank[9]. This method lends itself more readily to implementation in analog VLSI [6],[16]. Our approach using MRFs to segment color imagery is mathematically similar to the above surface reconstruction problems, with the exception that there are multiple surfaces representing each spectral component of the color imagery.

3. METHODS FOR COMPUTING COLOR DIFFERENCES

Of primary importance to the process of color segmentation is the use of color differencing methods. The manner in which significant changes in color are detected must play an important role in the formation of color change boundaries. The addition of color information holds promise for improving segmentation results since the color of imaged surfaces is more stable under geometric changes than the image irradiance [8]. Other studies have shown the usefulness of chromatic information in improving stereo matching algorithms [11].

Typically, a color image is composed of three spectral components obtained using filters with different sensitivities in the red, green, and blue wavelengths. In many cases, it is desirable to transform from the resulting red, green, and blue (RGB) images to other color spaces which separate intensity from color (hue). Kender's [12] discussion of color transformations points out that linear transformations from RGB data (e.g. YIQ) are preferable. Non-linear transformations such as intensity, hue, and saturation (IHS) and normalized color suffer from non-removable singularities and spurious gaps in the color distribution. However, with these constraints in mind, non-linear transforms often

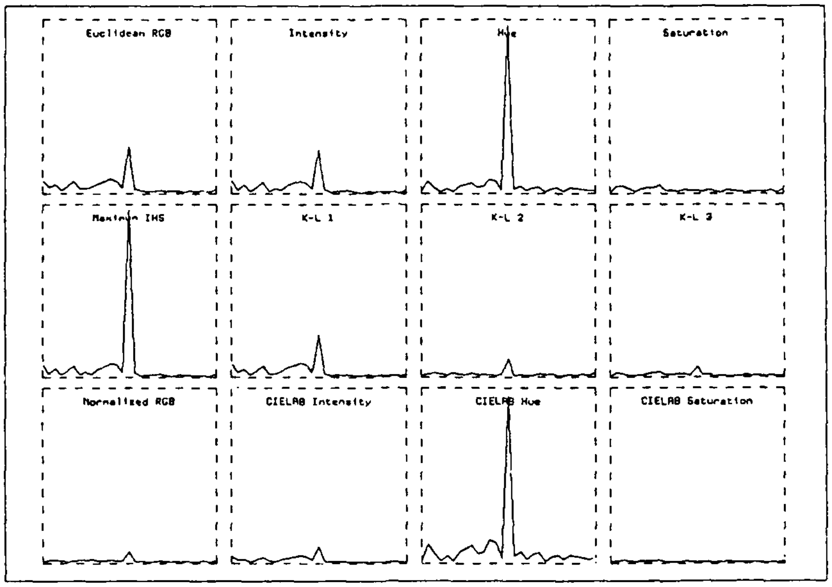
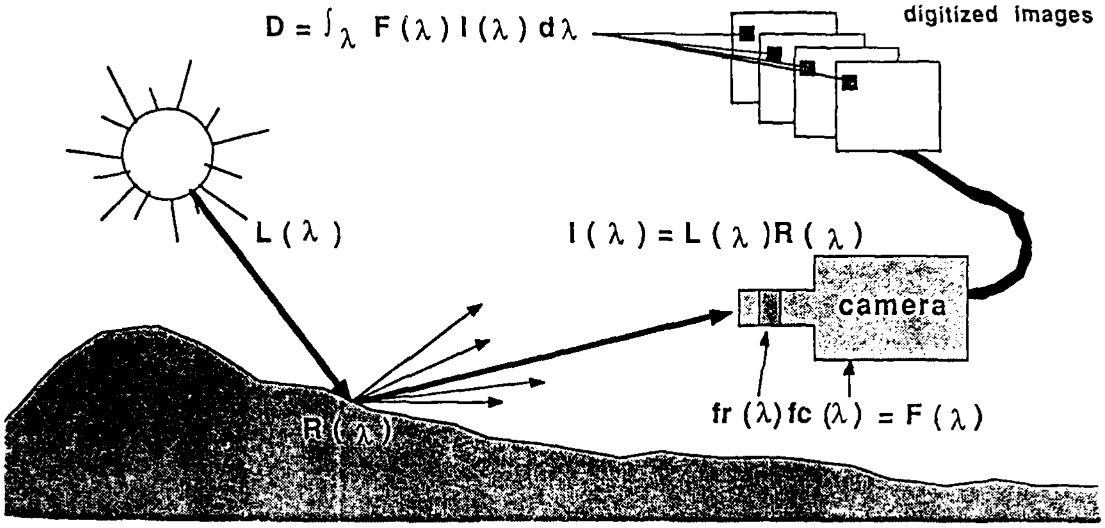


Figure 1. Graphs of the color difference between neighbors along the yellow line in Color Photo 1. The horizontal axis corresponds to distance along the yellow line, while the vertical axis signifies the magnitude of the color difference.

provide more useful information by separating intensity and hue information, as in the IHS system. For additional information on color transforms, see [19] and [23].

By comparing color differences computed using several transformations, we have found that measuring differences in hue consistently produces larger relative changes than other corresponding color and intensity measures normalized to the same range. The fact that hue tends to produce the difference with highest magnitude suggests the usefulness of color as a means of segmenting. However, in many instances, hue is also more susceptible to variation from noise, and, in practice, may not yield significantly better results (see Section 5.0). Figure 1 and Color Photo 1 show an experimental analysis of the usefulness of each measure in computing color differences. The graphs were obtained by normalizing and comparing the magnitude of the difference of each measure from the following set: ***Euclidean RGB, intensity, hue, saturation, Maximum IHS, CIELAB, CIELAB hue, CIELAB chroma, CIELAB intensity, Karhunen-Loeve components, and normalized color.*** Differences were computed between neighbors along the yellow line in Color Photo 1. For the Maximum IHS technique, the maximum difference in each component in IHS space is used as long as the intensity is above 10% (25 where 255 is the maximum). When the intensity is below 10%, only the intensity difference is considered.The CIELAB color transformation attempts to produce a uniform color space where color differences a human perceives as equal correspond to equal Euclidean distances [23]. The Karhunen-Loeve component [illegible] derived using the orthogonal Karhunen-Loeve expansion which minimizes the mean-square error in basis functions as well as a measure of dispersion (entropy function) for the three RGB components [22].

More recent work attempting to quantify the significance of a color change in an image is based on physical models of the sensors and environment [7],[8]. Briefly, the spectral response of a color sensor $F(\lambda)$ may be described as the product of a filter transmission function $f_r(\lambda)$ and the camera response characteristics $f_c(\lambda)$, as shown in Figure 2. The output of the sensor D is then the integrated product of the image irradiance $I(\lambda)$ and the sensor response $F(\lambda)$. Any number of color filters may be used to improve color discrimination capability. The image irradiance at a point, $I(\lambda)$, may be approximated by using an orthogonal set of basis functions and the sensor output D, as in [8]. Color differences between points are then computed by normalizing and calculating a distance between basis functions integrated over the entire visible spectrum, which gives more reliable results than measures operating at only one spectral wavelength. We have implemented this color differencing method, and our preliminary results indicate the use of a physical model for the sensor will greatly improve color differencing capability. Our future plans include making careful comparisons of the various transformations and differencing techniques and their effect on color segmentations using MRFs.



$D = \int_\lambda F(\lambda) I(\lambda) d\lambda$
digitized images
$L(\lambda)$
$I(\lambda) = L(\lambda) R(\lambda)$
camera
$R(\lambda)$
$fr(\lambda) fc(\lambda) = F(\lambda)$

Figure 2. Physically-based sensor model for color differencing metric. L(λ) is the spectral power distribution of the light. R(λ) is the reflectance function of the surface. F(λ) is the sensor response, while I(λ) is the image irradiance. D(λ) is the sensor output.

4. MRF LATTICES

4.1 Rectangular Lattices

The underlying lattice structure for the color and line processes plays an important role in the quality and structure of the segmentation. The simplest lattice structure is a rectangular grid, and the line processes are defined over some neighborhood of a pixel. By adjusting the energy function, larger cliques (neighbors of a site) encompassing additional directional and topographical structures can be heuristically added [3],[4]. For example, line process energies which penalize the formation of adjacent parallel discontinuities may be added. For the rectangular case, the line process can be broken into two components, horizontal and vertical, corresponding to the nearest neighbor, 4-connected system. Each of the four neighbors is of equal distance from the central pixel and is weighted identically.

In this paper, we have chosen to use the Hopfield net approach to minimization of the non-convex energy functionals, as in [16]. The sum of the following four energy terms, E_i, E_d, E_l, and E_g, must be minimized.

$$E_i = \lambda \sum_{i,j} ((\mathbf{f}_{i,j+1} \ominus \mathbf{f}_{i,j})^2 (1 - v_{i,j}) + (\mathbf{f}_{i+1,j} \ominus \mathbf{f}_{i,j})^2 (1 - h_{i,j}))$$

where E_i is the interpolation or smoothing term and λ controls the degree of smoothness in the interpolation term. The vector **f** represents the continuous-valued color process and is composed of red, green, and blue color components, and v and h correspond to the vertical and horizontal line processes respectively and vary continuously between 0 and 1. The symbol '$\ominus$' represents a color differencing scheme, such as described in Section 3. E_i is active when the line process variables v and h are less than one (i.e. the presence of a discontinuity is not yet certain). The energy data term is defined as

$$E_d = \alpha \sum_{i,j} ||\mathbf{f}_{i,j} - \mathbf{d}_{i,j}||^2$$

where the α parameter weights the importance of the input data, and **d** is a color process vector representing the observed color input image. This term ties the resulting segmentation to the original input. The third term, E_l, is the cost for introducing a new line process discontinuity and is defined as

$$E_l = \beta \sum_{i,j} (v_{i,j} + h_{i,j})$$

where β represents the penalty for formation of a line process discontinuity. When the difference terms in E_i become larger than the line process penalty β, it becomes cheaper to add a discontinuity than to continue smoothing. The final term forces the line processes to on (1) or off (0) states and is defined as follows:

$$E_g = \gamma \sum_{i,j} \left(\int_0^{v_{i,j}} g^{-1}(v)\, dv + \int_0^{h_{i,j}} g^{-1}(h)\, dh \right)$$

where g^{-1} is a standard sigmoid function representing the gain function for the line processes, and γ alters the gain term. The initial gain is nearly linear from 0 to 1 and, as the gain is increased, eventually becomes a step edge (see Color Photos 2 and 3), at which time the discontinuities are driven to their final states.

As in Hopfield's work [9], we chose the following update rule for the iterative minimization:

$$\frac{d\mathbf{f}_{i,j}}{dt} = \frac{-\partial E}{\partial \mathbf{f}_{i,j}}$$

$$\frac{dm_{i,j}}{dt} = \frac{-\partial E}{\partial v_{i,j}}$$

$$\frac{dn_{i,j}}{dt} = \frac{-\partial E}{\partial h_{i,j}}$$

where $E = E_i + E_d + E_l + E_g$, and m and n are internal state variables corresponding to the vertical and horizontal line processes respectively (i.e. $v_{i,j} = g(m_{i,j})$, and $h_{i,j} = g(n_{i,j})$). Finally, solving the above equations for the values of $m_{i,j}$, $n_{i,j}$, and $\mathbf{f}_{i,j}$, we obtain

$$m_{i,j} = \frac{\lambda(\mathbf{f}_{i,j+1} \ominus \mathbf{f}_{i,j})^2 - \beta}{\gamma}$$

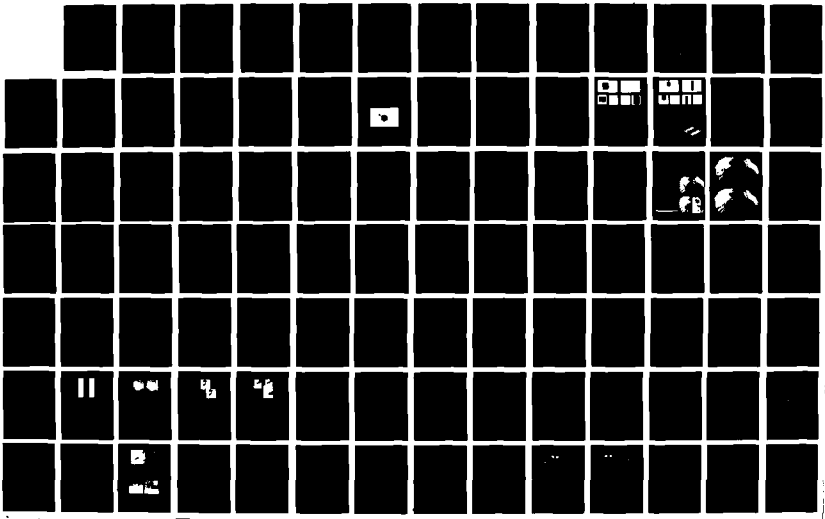
$$n_{i,j} = \frac{\lambda(\mathbf{f}_{i+1,j} \ominus \mathbf{f}_{i,j})^2 - \beta}{\gamma}$$

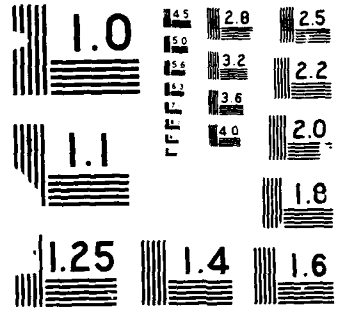
$$\mathbf{f}_{i,j} = \frac{\lambda(L^v_{i,j}\mathbf{f}_{i,j+1} + L^v_{i,j-1}\mathbf{f}_{i,j-1} + L^h_{i,j}\mathbf{f}_{i+1,j} + L^h_{i-1,j}\mathbf{f}_{i-1,j}) + \alpha \mathbf{d}_{i,j}}{\lambda(L^v_{i,j} + L^v_{i,j-1} + L^h_{i,j} + L^h_{i-1,j}) + \alpha}$$

where $L^v_{i,j} = (1 - v_{i,j})$ and $L^h_{i,j} = (1 - h_{i,j})$. In order to find the minimum energy solution, these coupled systems of equations must be iterated until convergence. In practice, we have found that updating the line process once every 10 color process updates is adequate.

Color Photo 3 shows a sequence of steps in the energy minimization for the image of natural terrain in Color Photo 2. The top row of Color Photo 3 is the current state of the segmented image. The second and third rows show the composite horizontal and vertical line processes and the gain function. From left to right, the gain is increasing, forcing the choice of discontinuities. The final segmentation is obtained after no change in the line or color process occurs for several increases in the gain function. For this example, we used a simple automatic method for setting the line process penalty β to the average neighborhood difference in color over the whole image. If fewer discontinuities and therefore fewer color regions are desired, β may be increased relative to the mean and standard deviation of the differences, while leaving the data weighting parameter α constant. The color differencing method used the maximum difference in IHS space, with hue and saturation values corresponding to low intensities being discarded. Color Photos 4 and 5 show results for different images on 128x128 MRFs using the same methods for parameter setting and a Euclidean RGB color differencing scheme. In Section 5, we address the difficult problems of choosing a color difference function and of determining parameter values.

AD-A212 981 PROCEEDINGS OF IMAGE UNDERSTANDING WORKSHOP HELD IN 7/13
PALO ALTO CALIFORNIA O..(U) SCIENCE APPLICATIONS
INTERNATIONAL CORP MCLEAN VA L S BAUMAN MAY 89
UNCLASSIFIED N00014-86-C-0700 F/G 17/11 NL




1.0
4.5
5.0
5.6
2.8
3.2
3.6
4.0
2.5
2.2
2.0
1.1
1.8
1.25
1.4
1.6

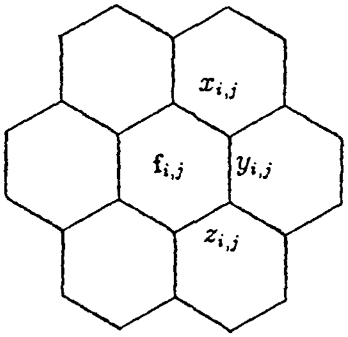


Figure 3. Structure of hexagonal lattice for MRF. The three line processes are labelled x, y, and z, and fall between the hexagonal sites. The color process vectors, $\mathbf{f}$, are located at the center of each hexagonal site and are composed of red, green, and blue components.

4.2 Hexagonal Lattices

Vertical and horizontal line processes on a rectangular lattice suffer from a bias toward rectilinear structures, as is evident in Color Photos 3, 4, and 5. For natural terrain imagery where 90° angles are rare, the effects are especially noticeable. In his book, *Robot Vision*, Horn gives three important advantages of hexagonal grids for computer vision [10]. Hexagonal grids allow improved sampling and quantization during the image formation process, markedly improve the understanding of connectivity (since all neighbors touch a center pixel), and allow easy use of edge detection masks (e.g. Laplacian masks). For our purposes, hexagonal lattices in an MRF allow increased resolution of the line process and replace the unnatural rectilinear bias with a hexagonal and triangular bias much more suitable to natural imagery. A given site in the hexagonal lattice has six equally distant neighbors and three different line process directions, each at 120° angles, as shown in Figure 3. In addition, modifying the energy terms used in the rectangular case is straightforward, as discussed below. The major disadvantage to using a hexagonal lattice is the reduction in resolution required to simulate hexagonal sampling from a rectangularly sampled image.* Increased accuracy in the hexagonal sampling can be achieved by increasing the size of the hexagons in the lattice and thus decreasing the effect of the digitization bias of the rectangular grid (see Color Photo 6).

The energy terms for the hexagonal case are similar to those for the rectangular lattice discussed earlier, with the exception that three line processes must be used, and special indexing for even and odd rows must be included. The new energy terms are:

$$E_i = \lambda \sum_{i,j} ((\mathbf{f}_{i-1,k} \ominus \mathbf{f}_{i,j})^2 (1 - x_{i,j}) + (\mathbf{f}_{i,j+1} \ominus \mathbf{f}_{i,j})^2 (1 - y_{i,j}) + (\mathbf{f}_{i+1,k} \ominus \mathbf{f}_{i,j})^2 (1 - z_{i,j}))$$

$$E_d = \alpha \sum_{i,j} ||\mathbf{f}_{i,j} - \mathbf{d}_{i,j}||^2$$

$$E_l = \beta \sum_{i,j} (x_{i,j} + y_{i,j} + z_{i,j})$$

$$E_g = \gamma \sum_{i,j} (\int_0^{x_{i,j}} g^{-1}(x)\, dx + \int_0^{y_{i,j}} g^{-1}(y)\, dy + \int_0^{z_{i,j}} g^{-1}(z)\, dz)$$

where x, y, and z are the line processes labelled in Figure 3. The subscript k is set based on whether the current site is on an even or odd row as follows:

$$k = j + 1 \quad (i \text{ even})$$

* This problem is purely an artifact of the rectangular grid CCD cameras used. A camera properly designed for computer vision research would use a hexagonal tesselation of photoreceptors.

$$k = j \quad (i \text{ odd})$$

As in the rectangular case, using Hopfield's update rule $d\mathbf{f}_{i,j}/dt = -\partial E/\partial \mathbf{f}_{i,j}$, $da_{i,j}/dt = -\partial E/\partial x_{i,j}$, $db_{i,j}/dt = -\partial E/\partial y_{i,j}$, and $dc_{i,j}/dt = -\partial E/\partial z_{i,j}$, we can solve for the values of $a_{i,j}$, $b_{i,j}$, $c_{i,j}$, and $\mathbf{f}_{i,j}$, which correspond to the three line processes, $x_{i,j}$, $y_{i,j}$, and $z_{i,j}$ and the color process, respectively. Let $L^x_{i,j} = (1 - x_{i,j})$, $L^y_{i,j} = (1 - y_{i,j})$, and $L^z_{i,j} = (1 - z_{i,j})$. Then the update rule for $\mathbf{f}_{i,j}$ is

$$\mathbf{f}_{i,j} = \frac{\lambda(L^x_{i,j}\mathbf{f}_{i-1,s1} + L^x_{i+1,s2}\mathbf{f}_{i+1,s2} + L^y_{i,j}\mathbf{f}_{i,j+1} + L^y_{i,j-1}\mathbf{f}_{i,j-1} + L^z_{i,j}\mathbf{f}_{i+1,s1} + L^z_{i-1,s2}\mathbf{f}_{i-1,s2}) + \alpha d_{i,j}}{\lambda(L^x_{i,j} + L^x_{i+1,s2} + L^y_{i,j} + L^y_{i,j-1} + L^z_{i,j} + L^z_{i-1,s2}) + \alpha}$$

$$a_{i,j} = \frac{\lambda(\mathbf{f}_{i-1,k} \ominus \mathbf{f}_{i,j})^2 - \beta}{\gamma}$$

$$b_{i,j} = \frac{\lambda(\mathbf{f}_{i,j+1} \ominus \mathbf{f}_{i,j})^2 - \beta}{\gamma}$$

$$c_{i,j} = \frac{\lambda(\mathbf{f}_{i+1,k} \ominus \mathbf{f}_{i,j})^2 - \beta}{\gamma}$$

where s1 and s2 are defined as

$$s1 = j + 1, \ s2 = j \quad (i \text{ even})$$

$$s1 = j, \ s2 = j - 1 \quad (i \text{ odd})$$

and k is defined as above. Again, the symbol '$\ominus$' represents a color differencing scheme. Color Photo 7 shows a comparison of the results for the rectangular and hexagonal grids applied to the same image with identical line process penalty values and using the maximum IHS difference, as mentioned in Section 3. Both MRFs were computed on 35x40 lattices (so hexagons have approximately 14 pixel width) sub-sampled from the original 512x512 image and, using a Symbolics 3650, required approximately 15 and 20 minutes respectively to converge. The discontinuities for the hexagonal method fit the natural terrain much better than the rectangular method. Color Photo 8 shows the results for a 64x64 hexagonal MRF applied on the natural terrain image of Color Photo 2, with similar parameters and using the Euclidean RGB metric.

4.3 Other Lattices

There are only three possible tesselations of the image plane using a regular polygon: a square, a hexagon, and a triangle. In addition to the square and hexagonal cases, we have investigated the use of triangular lattices for image segmentation. A triangular lattice can be used to define the color and line processes in a similar fashion to the hexagonal method. In fact, the triangular lattice actually forms a hexagonal lattice of lower resolution. The triangular lattice requires three line processes, and each site has only three equidistant neighbors along its sides. A potential advantage over the hexagonal and rectangular methods is that at each vertex in the triangular lattice, six possible directions for the discontinuity exist, while only three and four exist for the hexagonal and rectangular lattices, respectively. However, to date, results for the triangular lattice do not appear significantly different than those for the hexagonal technique and require additional complexity to implement. Properly indexing neighbors and each line process requires even and odd rows *and* columns to be addressed differently. In addition, convergence is painfully slow and tends to oscillate. We are currently investigating a formulation for continuous line processes that may allow more freedom in the choice of line process direction and location.

5. MRF PARAMETERS

Efficient methods of automatically setting parameters in the MRF segmentation algorithm will increase the overall usefulness of the method. There are four parameters of importance mentioned in Section 4. λ controls the degree of smoothing in the interpolation term. Higher values of λ increase the amount of smoothing. In our experiments with parameters, the value of λ is kept constant at one. The line process penalty, β, penalizes the formation of color discontinuities. Higher values of β will restrict the formation of line process discontinuities and thereby allow increased smoothing of the color process. The data weighting term, α, allows control over the importance of the input data. Higher values of α place more confidence in the validity of the input color image, while lower values are desirable for noisy data. The gain energy term is altered using the parameter γ, which we set equal to one for all of our parameter experiments. Larger values of γ favor the current state of the line processes.

Of particular importance in parameter estimation is the ratio of α to β. By increasing this ratio, a higher density of color discontinuities will form, while lowering the ratio produces fewer discontinuities and increases smoothing. We chose to test the effect of setting $\alpha = 0.1$ and varying β over the entire range of possible values. Figure 4 shows the effect of the line process penalty β on the percentage or density of discontinuities for Color Photo 6 using a 64x64 hexagonal lattice in the MRF. The color differencing method was Euclidean RGB normalized to the range from 0 to 255. The percentage of discontinuities was computed for each integer value of β after the network had reached a final solution. This required computation of 256 separate MRF segmentations, one for each value of β. To speed up the process of testing the parameter values, we developed a parallel implementation using 8 Lisp machines on an Ethernet. The results still required over 60 hours of computation. Fortunately, MRF algorithms are easily parallelized and are ideal for analog VLSI implementations or fine grained parallel architectures like the Connection Machine.

As shown in Figure 4, when β was zero, the final state of the line process consisted of 100% discontinuities, and no smoothing of the input data occured (i.e. the original image was the result). When β was equal to the maximum possible color difference of 255, no discontinuities formed and the result was merely a smoothed version of the input data. We have computed similar graphs for several different color images and found that the inverse logarithmic relationship of the curve between β and the percentage of discontinuities holds, with minor shifts in the location of the "elbow". Subjectively, the best segmentations appear to occur near this levelling in the density of discontinuities.

Of interest in Figure 4 is a peculiar stabilization and sudden drop in the density of the line process discontinuities at several intervals. These sudden drops in the percentage of discontinuities do not appear to be an artifact of either the algorithms or the accuracy of the computer, but rather a result of the interaction between the underlying lattice structure and the image content. At extremely high resolution of the line process penalty, the sudden drops are equally as abrupt with no evidence of levelling out. We are currently exploring possible explanations. Perhaps as the line process penalty changes, lower energy states will correspond to shapes that best match the underlying lattice structure at discontinuous locations.

In fact, the line process penalty may be used as a kind of "temperature" to perform annealing instead of increasing the gain of the sigmoid function which is set to a constant value. The value of β is lowered from the maximum penalty to zero. When the change in the current state for a given value of β falls below a small threshold, β is decreased by some increment (one in this case). Each time the value of β is lowered, the density of discontinuities is plotted, resulting in a similar graph shown in Figure 5. Viewed from the perspective of statistical physics, the unchanging portions of the curve may correspond to "equilibrium states", while at critical temperatures the sudden addition of new line discontinuities may be similar to "phase transitions" in metals, ferromagnets, and ideal gases (see [4] and [13]).

We have also experimented with changing the parameters locally rather than globally over the image. For instance, the weighting term for the input data, α, may be given values which vary over the image corresponding to the expectation of noise in localized regions. Higher values of α place more confidence in the reliability of color information, whereas lower values indicate a low signal to noise ratio. While implementation of this scheme is simple, the major drawback to allowing parameters to vary locally is the incredible increase in difficulty of parameter understanding. The

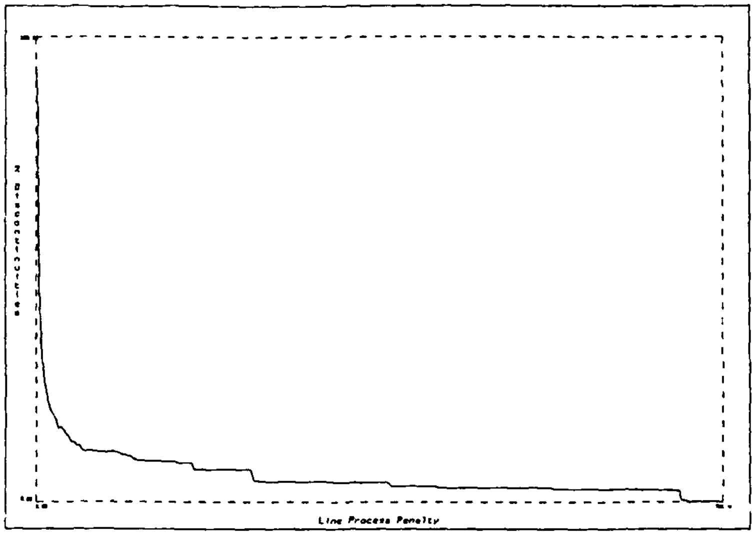


Figure 4. The effect of the line process penalty β (horizontal axis) on the percentage of discontinuities (vertical axis). For each integral value of β, the final state of the line process is computed.

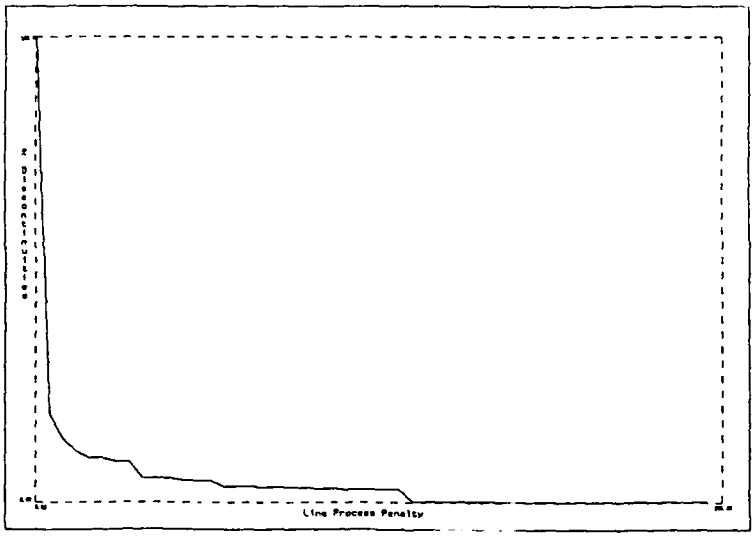


Figure 5. Intermediate states of the percentage of line proces ontinuities at successively lower values of β for a single MRF.

interrelationships of variable parameters make it difficult to predict the effects of changing parameters on the global segmentation result.

Another important issue in understanding and controlling segmentation results depends upon the choice of color difference functions, as mentioned in Section 3. We have only begun to analyze the effects of various differencing methods on the final segmentations. A comparison of the results for the Euclidean RGB metric and the Maximum IHS method described in Section 3 is shown in Color Photo 9. All other parameters are equal for the two results. The IHS method is clearly more sensitive to color changes for equal line process penalty values and caused excessive fragmentation. In the near future, we plan to use the color metric based on the sensor response discussed briefly in Section 3 and compare its results to existing techniques.

6. CONCLUSIONS

We have shown results for color segmentation of natural terrain imagery using MRFs. Comparisons of the results for both square and hexagonal lattices using color differencing methods have been presented. We have also discussed ongoing research in the use of color metrics and in understanding MRF parameters.

We intend that the methods used in this paper for color segmentation be included in a larger perception system capable of recognizing various objects in natural terrain such as rocks, trees, bushes, gullies, and other types of vegetation. By producing an accurate description of the color of image regions which is both region-based (color process) and boundary-based (line process) and integrating additional visual cues from laser range data and other sensors (perhaps by coupled MRFs, as in [3]), we expect to produce more robust identification and therefore improve corresponding autonomous navigation capabilities. However, implementations of MRF algorithms will require specialized hardware to be practical for online use in robotic perception systems.

Acknowledgements

The author thanks John Harris for stimulating thought on the subject and the AI Center staff for their continued support.

References

[1] Chou, P. and C. Brown, "Multimodal Reconstruction and Segmentation with Markov Random Fields and HCF Optimization," *Proc. of the DARPA IU Workshop*, Cambridge, MA, pp. 214-221, April, 1988.

[2] Daily, M.J., J.G. Harris, K.E. Olin, K. Reiser, D.Y. Tseng, and F.M. Vilnrotter, "Knowledge-based Vision Techniques Annual Technical Report," U.S. Army ETL, Fort Belvoir, VA, October, 1987.

[3] Gamble, E. and T. Poggio, "Visual Integration and the Detection of Discontinuities: the Key Role of Intensity Edges," MIT AI Lab, A.I. Memo 970, October, 1987.

[4] Geman, S. and D. Geman, "Stochastic Relaxation, Gibbs Distributions, and the Bayesian Restoration of Images," *IEEE Trans. on PAMI*, Vol. PAMI-6, pp. 721-741, November, 1984.

[5] Hanson, A.R. and E.M. Riseman, "Segmentation of Natural Scenes," in *Computer Vision Systems*, (A.Hanson and E.Riseman, Eds.), Academic Press, Orlando, FL, pp. 129-163, 1978.

[6] Harris, J.G. and C. Koch, "Resistive Fuses: Circuit Implementations of Line Discontinuities in Vision," to be submitted to the 1989 Snowbird Neural Network Meeting, 1989.

[7] Healey, G. and T. Binford, "The Role and Use of Color in a General Vision System," *Proc. of the DARPA IU Workshop*, Los Angeles, CA, pp. 599-613, February, 1987.

[8] Healey, G. and T. Binford, "A Color Metric for Computer Vision," *Proc. of the DARPA IU Workshop*, Cambridge, MA, pp. 854-861, April, 1988.

[9] Hopfield, J.J. and D.W. Tank, "Neural Computation of Decisions in Optimization Problems," *Biological Cybernetics*, Vol. 52, No. 14, pp. 152, 1985.

[10] Horn, B.K.P., *Robot Vision*, MIT Press, Cambridge, MA, 1986.

[11] Jordan, J.R. and A.C. Bovik, "Computational Stereo Vision Using Color," *IEEE Control Systems Magazine*, pp. 31-36, June, 1988.

[12] Kender, J., "Saturation, Hue, and Normalized Color: Calculation, Digitization Effects, and Use," Technical Report, Department of Computer Science, Carnegie-Mellon University, 1976.

[13] Kirkpatrick, S. and R.H. Swendsen, "Statistical Mechanics and Disordered Systems," *Comm. of the ACM*, Vol. 28, No. 4, pp. 363-373, April 1985.

[14] Klinker, G.J., S.A. Shafer, and T. Kanade, "The Measurement of Highlights in Color Images," *International Journal of Computer Vision*, Vol. 2, No. 1, pp. 7-32, June, 1988.

[15] Klinker, G.J., S.A. Shafer, and T. Kanade, "Image Segmentation and Reflection Analysis Through Color," *Proc. of the DARPA IU Workshop*, Cambridge, MA, pp. 838-853, April, 1988.

[16] Koch, C., J. Marroquin, and A. Yuille, "Analog 'Neuronal' Networks in Early Vision," MIT AI Lab, AI Memo 751, June, 1985.

[17] Marroquin, J. L., "Probabilistic Solutions of Inverse Problems," AI-TR 860, MIT AI Lab, September, 1985.

[18] Ohlander, R., K. Price, and D.R. Reddy, "Picture Segmentation using a Recursive Region Splitting Method," *Computer Graphics and Image Processing*, Vol. 8, pp. 313-333, 1978.

[19] Ohta, Y., T. Kanade, and T. Sakai, "Color Information for Region Segmentation," *Computer Graphics and Image Processing*, Vol. 13, pp. 222-231, 1980.

[20] Poggio, T., E. Gamble, and J. Little, "Parallel Integration of Vision Modules," *Science*, Vol 242, pp. 436-438, 1988.

[21] Thorpe, C., S. Shafer, and T. Kanade, "Vision and Navigation for the Carneige Mellon Navlab," *Proc. of the DARPA IU Workshop*, Los Angeles, CA, pp. 143-152, February, 1987.

[22] Tou, J.T. and R.C. Gonzalez, *Pattern Recognition Principles*, Addison-Wesley, Reading, MA, 1981.

[23] Wyszecki, G. and W.S. Stiles, *Color Science: Concepts and Methods, Quantitative Data and Formulae*, John Wiley & Sons, New York, NY, 1982.

See Reference section on page 1147 for three pages of color photographs.

Extracting Shape and Reflectance of Hybrid Surfaces by Photometric Sampling

Shree K. Nayar, Katsushi Ikeuchi, and Takeo Kanade

The Robotics Institute, Carnegie Mellon University, Pittsburgh, Pennsylvania 15213

Abstract

Machine vision can greatly benefit from the development and utilization of accurate reflectance models. The hybrid reflectance model is a linear comt i:ation of Lambertian and specular components and is obtained from the primary reflection components by making surface assumptions. Hence, Lambertian and specular surfaces are special instances of hybrid surfaces. We present a method for determining the shapes of hybrid surfaces without prior knowledge of the relative strengths of the Lambertian and specular components of reflection. The object surface is illuminated using extended light sources and is viewed from a single direction. Surface illumination using extended sources makes it possible to ensure the detection of both Lambertian and specular reflections. Uniformly distributed source directions are used to obtain an image sequence of the object. This method of obtaining photometric measurements is called photometric sampling. An extraction algorithm uses the set of image intensity values measured at each surface point to compute orientation as well as relative strengths of the Lambertian and specular reflection components. Experiments were conducted on Lambertian surfaces, specular surfaces, and hybrid surfaces. The results reflect a high accuracy in measured orientations and estimated reflectance parameters.

1 Introduction

Most machine vision problems involve the analysis of images resulting from the reflection of light. The apparent brightness of a point depends on its ability to reflect incident light in the direction of the sensor; what is commonly known as its reflectance properties. Therefore, the prediction or interpretation of image intensities requires a sound understanding of the various mechanisms involved in the reflection process. While shape extraction methods are being developed and refined, it is also essential for the vision community to research and utilize more sophisticated reflectance models. Once a "general" reflectance model is made available, we will be free to make reflectance assumptions that are appropriate for the vision application at hand. The result, a more specific model, may then be used to develop efficient shape extraction techniques.

Shape from shading [4][6][13], photometric stereo [18][7][2], and local shape from specularity [3] are examples of techniques that extract three-dimensional shape information from photometric measurements. All of these techniques rely on prior knowledge of surface reflectance properties. The reflectance properties are either assumed or measured using a calibration object of known shape. In many real-world applications, such as those involving surfaces of different reflectance characteristics, the calibration approach is not a practical one. Therefore, the existing shape extraction methods are often used by assuming surface reflectance to be either Lambertian or specular. Many surfaces encountered in practice are hybrid in reflectance, that is, their reflectance models are linear combinations of Lambertian and specular models. Therefore, Lambertian and specular models are only limiting instances of the hybrid model. It is desirable to have a method that is capable of extracting the shape of hybrid surfaces, including Lambertian and specular ones.

In many industrial applications, surface pol'sh and roughness are found to be important inspection criteria. In such cases, surface reflectance properties may be interpreted as measures of surface polish and roughness. Furthermore, reflectance properties may be used to segment an image into different regions; each region may then be regarded as a different surface to aid the process of inspection. For these reasons, it would be of great value to have a technique that could, in addition to determining shape, also estimate the reflectance properties of each surface point.

We begin this paper with a summary of the various mechanisms involved in the reflection process. By considering both physical optics and geometrical optics approaches, the primary components of reflection are identified. By making assumptions related to the microscopic shape of surfaces, the primary reflection components are simplified to obtain the hybrid reflectance model. The object of interest is illuminated using extended light sources and is viewed from a single direction. The sources are uniformly distributed around the object and are sequentially scanned to obtain an image sequence of the object. We refer to this method of obtaining photometric measurements as *photometric sampling*. An extraction algorithm uses the image sequence and the hybrid reflectance model to determine object shape. Shape information is extracted without prior knowledge of the relative strengths of the Lambertian and specular reflection components. In addition, the extraction algorithm also estimates reflectance parameters at surface points.

2 Surface Reflection

2.1 A Unified Perspective

When light is incident on a boundary interface between two different media, it is reflected according to well-known laws. There are two different approaches to optics and, consequently, two different approaches to the study of reflection. *Physical*, or *wave* optics, is based directly on electromagnetic wave theory and uses Maxwell's equations to study the propagation of light. *Geometrical*, or *ray* optics, on the other hand, uses the short wavelength of light to simplify many of the light propagation problems. While geometrical reflectance models may be construed as mere approximations to physical reflectance models, they possess simpler mathematical forms that often render them more usable than physical models. However, geometrical models are applicable only when the wavelength of incident light is small when compared to the dimensions of the *microscopic* surface imperfections. Therefore, it is incorrect to use these models to interpret or predict reflections from smooth surfaces; only physical models are capable of describing the underlying reflection mechanism.

In [11] we have unified physical and geometrical approaches to describe reflection from surfaces that may vary from smooth to rough. More specifically, we have focussed on the Beckmann-Spizzichino (physical optics) model and the Torrance-Sparrow (geometrical optics) model. The surface height is modeled as a continuous stationary random process with standard deviation, σ_h, representing the physical *roughness* of the surface (Figure 1). The spatial variation of the surface is described in terms of a correlation function that determines the dependence between the heights of different points on the surface. The spatial frequency of the surface is represented by the correlation distance, T. The incident light is assumed to be a plane electromagnetic wave with wavelength, λ. The reflectance curves predicted by the physical and geometrical models are obtained by varying the three parameters σ_h, T, and λ. From studying the behaviors of the physical and geometrical optics models, it is seen that surface radiance may be decomposed into three primary reflection components, namely, the *diffuse lobe*, the *specular lobe*, and the *specular spike*.

The diffuse component results from two main mechanisms. In one case, light rays that impinge on the surface are reflected many times between surface undulations before they are scattered into space. If these *multiple reflections* occur in a random manner, the incident energy is distributed in all directions, resulting in diffuse reflection. Another mechanism leading to diffuse reflection is the *internal scattering*[1] of light rays. In this

[1] This mechanism is often referred to as "body" reflection.

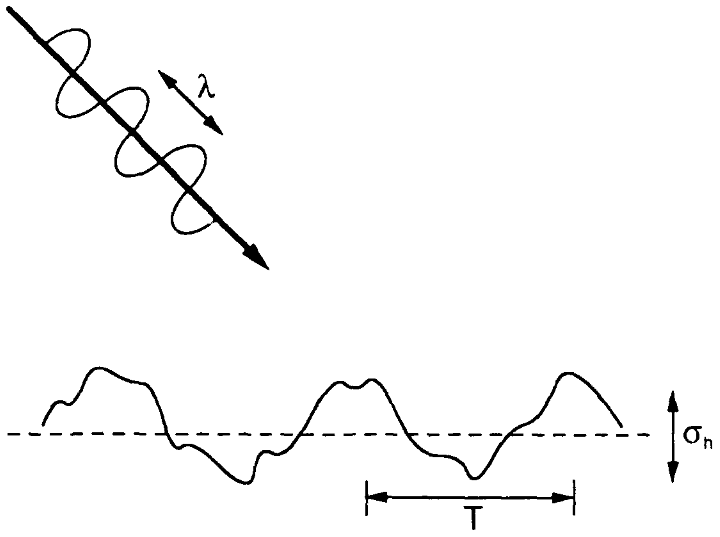


Figure 1: Surface Reflection is closely related to the microscopic surface profile and the wavelength of incident light.

case, the light rays penetrate the surface and encounter microscopic inhomogeneities in the surface medium. The light rays are repeatedly reflected and refracted at boundaries between regions of differing refractive indices. Some of the scattered rays find their way back to the surface in a variety of directions, resulting in diffuse reflection.

Specular reflection is composed of two primary components: the specular lobe and the specular spike. The lobe component spreads around the specular direction, and the spike component is zero in all directions except for a very narrow range around the specular direction. The relative strengths of the two components are dependent on the microscopic roughness of the surface. A detailed analysis of the characteristics of the three reflection components is given in [11]. We summarize our findings with the following remarks:

- The diffuse component may be represented by the Lambertian model [9]. This model has been used extensively to test shape-from-shading and photometric stereo techniques, and the results have indicated that it performs reasonably well. More accurate models [8] [14] may be used at the cost of functional complexity.
- The Beckmann-Spizzichino physical optics model predicts both the specular lobe and spike components. For a very smooth surface ($\sigma_h \ll \lambda$), the spike component dominates and the surface behaves like a mirror. As the roughness increases, however, the spike component shrinks rapidly, and the lobe component begins to dominate. The two components are simultaneously significant for only a small range of roughness values.
- A "sharp" specular component may result from the specular spike component when the surface is smooth ($\sigma_h/\lambda < 1.5$) *and/or* the specular lobe component when the surface is gently undulating ($\sigma_h/T < 0.02$).
- The Torrance-Sparrow geometrical optics model provides a good approximation of the specular lobe component of the Beckmann-Spizzichino model. Both models are successful in predicting *off-specular* peaks

in the specular lobe component. Due to its simpler mathematical form, the Torrance-Sparrow model may be used to represent the specular lobe component.

- The Torrance-Sparrow model is not capable of describing the mirror-like behavior of smooth surfaces, and it should not be used to represent the specular spike component as it would produce erroneous results.
- The specular lobes of both Torrance-Sparrow and Beckmann-Spizzichino models tend to have *specular peaks*, rather than off-specular peaks, when the viewing direction is fixed and the source direction is varied.

In shape extraction techniques such as photometric stereo and structured highlight, images of the observed object are obtained by varying the source direction while keeping the viewing direction constant. The shape extraction method described in this paper is also based on the same approach. The shapes and functional forms of individual reflection components are different for the varying viewing direction and varying source direction cases. We emphasize this difference by introducing a new representation of the reflection components. Figure 2 shows polar plots of the diffuse lobe, specular lobe, and specular spike. The magnitudes of the three components of the radiance value in the viewer direction are determined by intersections made by the lobes with the line joining the source and the origin. The diffuse component is represented by the Lambertian model, specular lobe component by the Torrance-Sparrow model, and the specular spike component by the spike component of the Beckmann-Spizzichino model.

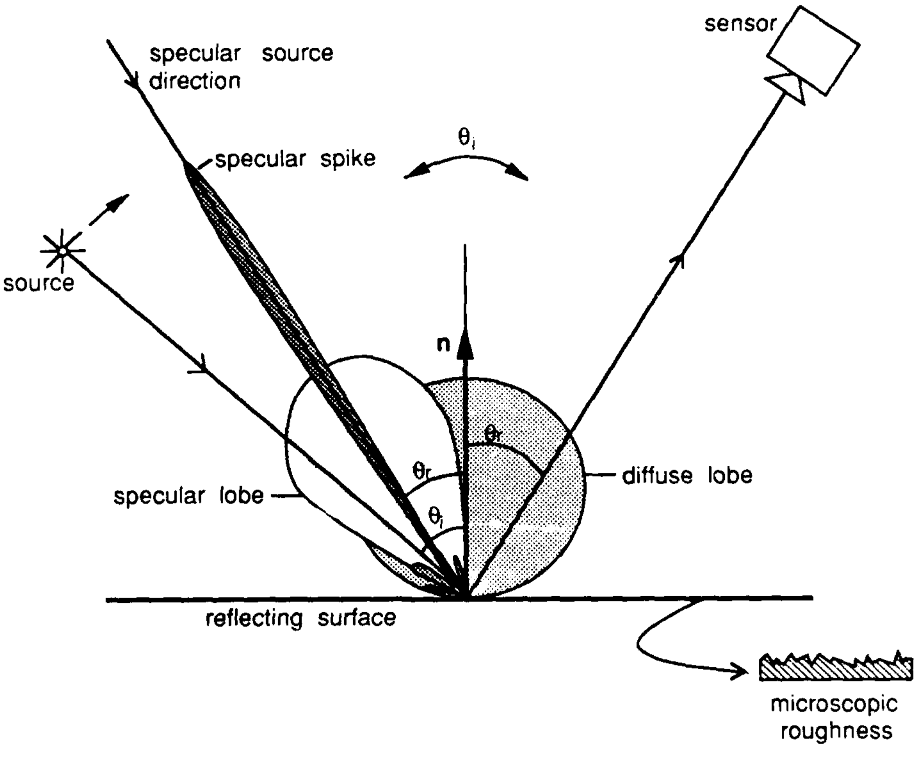


Figure 2: Polar plots of the primary reflection components as functions of the source angle for a given viewing direction.

2.2 Hybr·' Reflectance Model

In this paper, we assume that the surfaces of interest are smooth, i.e. either the surface roughness is comparable to the wavelength of incident light ($\sigma_h/\lambda < 1.5$), or the surface is gently undulating ($\sigma_h/T < 0.02$), or both. From the previous discussion we see that, under these conditions, both the spike and lobe components can be significant only in a narrow region around the specular direction. Therefore, we will combine the spike and lobe components into a single component, namely the *specular component*. We also assume that the surfaces under consideration are non-homogeneous. Therefore, a diffuse component of reflection may result from the internal scattering mechanism. We use the Lambertian model to represent the diffuse component. The combination of the above mentioned two components is referred to as the *hybrid reflectance model*.

Consider the illumination of an object by a point source of light, as shown in Figure 3. The point source emits light in all directions. Light energy reflected by the surface in the direction of the camera causes an image of the surface to be formed. The intensity at any point in the image of the surface may be expressed as:

$$I = IL + IS, \tag{1}$$

where IL is the Lambertian intensity component and IS is the specular intensity compenent.

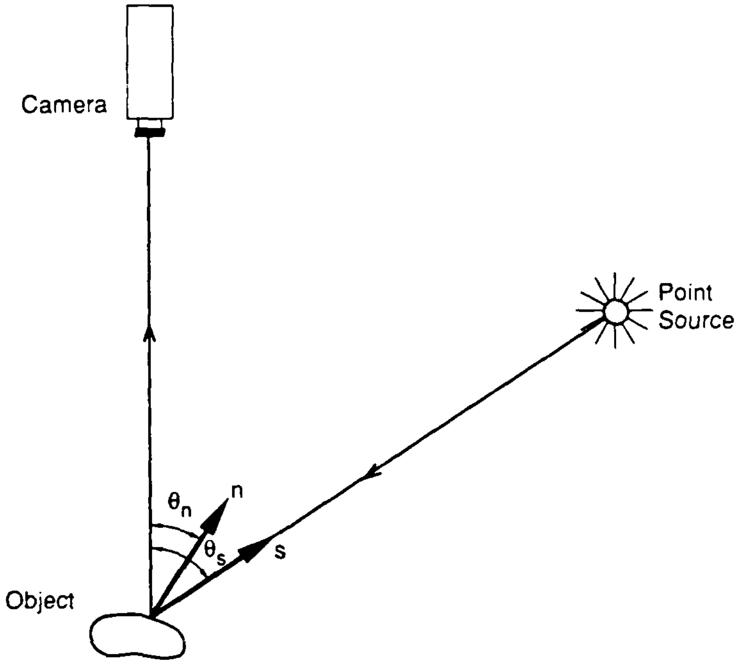


Figure 3: Two-dimensional illumination and imaging geometry. A surface element with orientation θ_n reflects light from the point source direction θ_s into the camera.

We will express the two components of image intensity in terms of the parameters that describe the two-dimensional imaging and illumination geometry shown in Figure 3. In two dimensions, the source direction vector **s**, surface normal vector **n**, and viewing direction vector **v** lie in the same plane. Therefore, directions are represented by a sing' ,parameter, namely, the zenith angle θ.

Lambertian Component: The brightness of a Lambertian surface is proportional to the energy of incident light. As can be seen in Figure 3, the amount of light energy falling on a surface element is proportional to the area of the surface element as seen from the source position, often referred to as the foreshortened area. The foreshortened area is a cosine function of the angle between the surface orientation direction θ_n and the source direction θ_s. Therefore, the Lambertian intensity component IL may be written as:

$$IL = A\,cos(\theta_s - \theta_n)\,, \tag{2}$$

where the constant A determines the fraction of incident energy that is diffusely reflected. We have assumed that the angle of incidence, $(\theta_s - \theta_n)$, is greater than $-\pi/2$ and less than $\pi/2$, i.e. IL is always greater than zero.

Specular Component: Since the specular intensity component IS is a very sharp function of the source direction, it may be approximated by the delta function [11]:

$$IS = B\,\delta(\theta_s - 2\theta_n)\,. \tag{3}$$

The *basic photometric function*[2] relates image intensity to surface orientation, surface reflectance, and point source direction and may be written by substituting equations 2 and 3 into equation 1 to obtain:

$$I = A\,cos(\theta_s - \theta_n) + B\,\delta(\theta_s - 2\theta_n)\,. \tag{4}$$

The constants A and B in equation 4 represent the relative strengths of the Lambertian and specular components of reflection, respectively. We call A and B the reflectance parameters. We see that $A > 0$ and $B = 0$ for a purely Lambertian surface, $A = 0$ and $B > 0$ for a purely specular surface, and $A > 0$ and $B > 0$ in general.

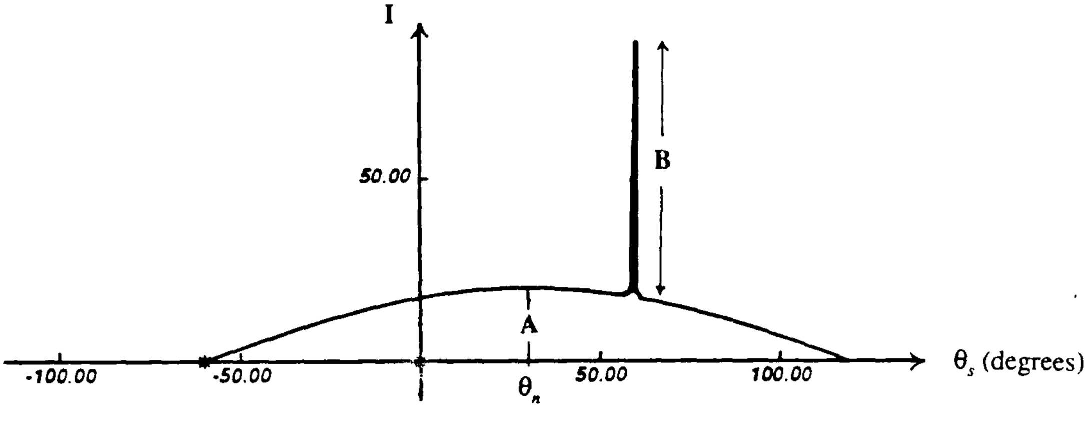


Figure 4: Basic photometric function $I(\theta_s)$ for a hybrid surface.

Our objective is to determine orientation and reflectance at each surface point from a set of image intensities that result from changing the source direction θ_s. Therefore, we will often refer to the basic photometric function as $I(\theta_s)$, a relation between image intensity and source direction. Figure 4 shows a plot of the basic photometric function for a hybrid surface of given orientation.

[2] The photometric function is similar to the image irradiance [5] eq'ation, since image intensity is assumed to be proportional to image irradiance.

3 Photometric Sampling Using Extended Sources

3.1 Why Extended Sources ?

We propose to illuminate the object surface by using extended sources, rather than point sources, for the following reasons:

- In the case of point source illumination, specular reflection is not detected unless $\theta_s = 2\theta_n$. In order to determine shape and reflectance parameters of specular and hybrid surfaces, specular reflections must be captured in the measured intensities. To detect specular reflections from surface points of all orientations, an infinite number of point sources need to be positioned around the surface. Such an approach is unrealistic from the perspective of practical implementation. Unlike a point source, an extended source emits light from an area of points rather than a single point. Therefore, a small number of extended sources may be used to ensure the detection of specular reflections.
- In the case of point source illumination, image intensities due to specular reflections are often observed to be much greater than intensities resulting from Lambertian reflections [15]. Therefore, it is difficult to measure both components in the same image. Extended source illumination tends to make the image intensities due to Lambertian and specular reflections comparable to one another. A specular surface element of a given orientation will reflect light from a small area on the extended source into the camera. On the other hand, a Lambertian surface element of the same orientation reflects light from all points on the extended source. This feature of the proposed illumination scheme makes it possible to measure both Lambertian and specular reflections in the same image.

In Appendix A, we have shown how extended sources are generated. The extended source radiance function, $L(\theta, \theta_s)$, is derived, and the parameters that determine the direction and limits of an extended source are defined. These results will be extensively used in the following discussions.

3.2 Photometric Function for Extended Sources

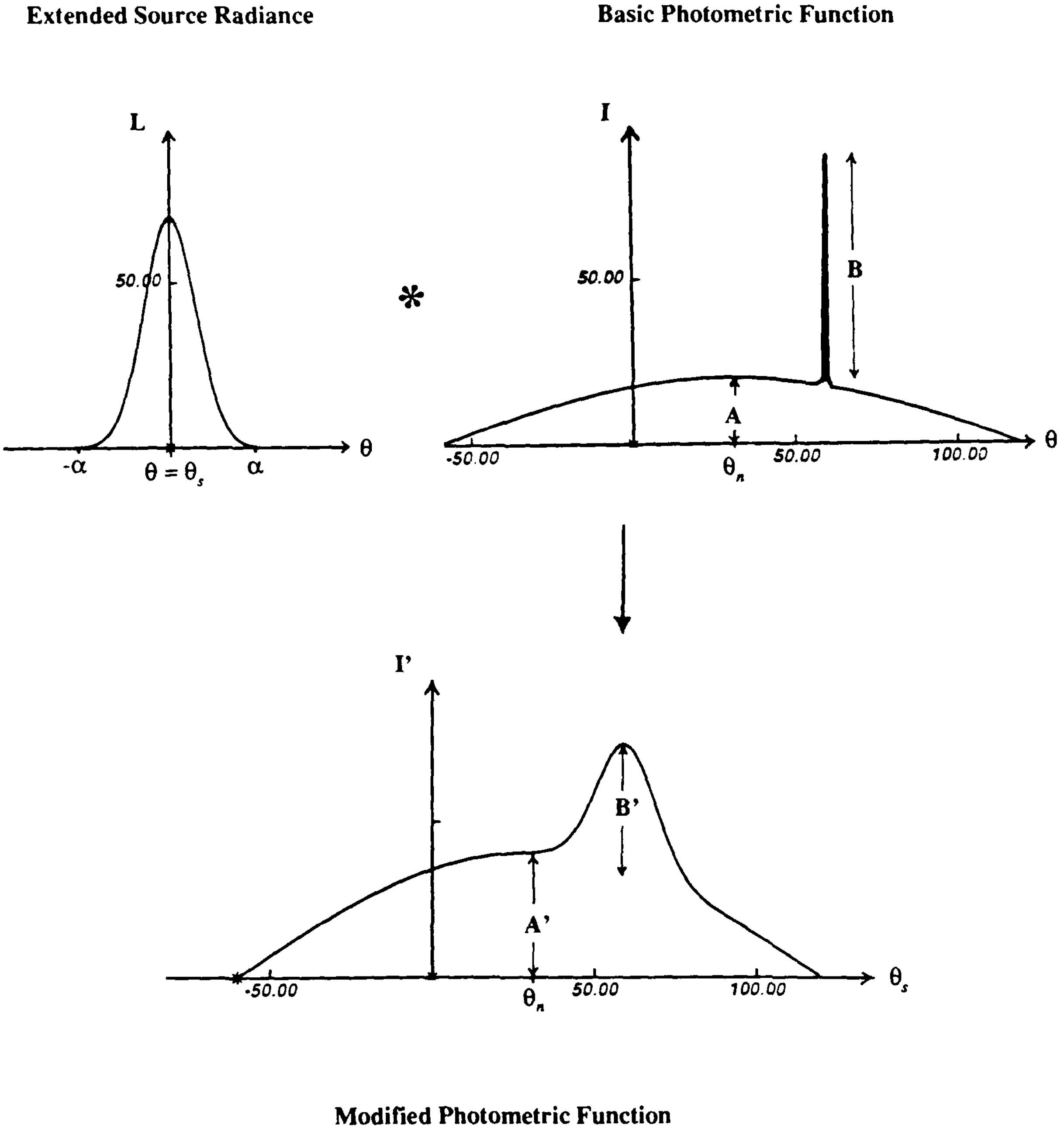
The photometric function for point source illumination (equation 4) needs to be modified for extended source illumination. An extended source may be thought of as a collection of point sources in which each point source has a radiant intensity that is dependent on its position on the extended source. The intensity of light reflected by a surface may be determined by computing the integral of the light energies reflected from all points on the extended source. Therefore, the modified photometric function $I'(\theta_s)$ is determined by convolving the basic photometric function $I(\theta)$ with the extended source radiance function $L(\theta, \theta_s)$. This operation is illustrated in Figure 5. For a surface point of orientation θ_n, the Lambertian component IL' of the modified photometric function is determined as:

$$IL' = A \int_{\theta_s-\alpha}^{\theta_s+\alpha} L(\theta, \theta_s) \cos(\theta - \theta_n)\, d\theta \,. \tag{5}$$

The limits of the integral are determined by the width of the extended source (Appendix A). It can be shown [10] that IL' is a cosine function of the angle between the surface orientation and the direction corresponding to the "center of mass" of the extended source radiance distribution, $L(\theta, \theta_s)$. In our case, since the extended source is symmetric with respect to the source direction θ_s, the center of mass of the radiance function is in the direction θ_s. Therefore, we obtain:

$$IL' = A' \cos(\theta_s - \theta_n) \,, \tag{6}$$

where the constant A' represents the strength of the Lambertian component.



Extended Source Radiance

Basic Photometric Function

Modified Photometric Function

Figure 5: The photometric function for extended source illumination is obtained by convolving the basic photometric function with the extended source radiance function.

Similarly, the specular intensity component IS' resulting from the extended source $L(\theta, \theta_s)$ is determined as:

$$IS' = B \int_{\theta_s - \alpha}^{\theta_s + \alpha} L(\theta, \theta_s)\, \delta(\theta - 2\theta_n)\, d\theta \,, \tag{7}$$

or:

$$IS' = B\, L(2\theta_n, \theta_s) \,. \tag{8}$$

Strictly speaking, the result of the above integral is dependent on the exact shape of the specular spike and lobe. However, since both components are significant only in the specular direction, $2\theta_n$, it is reasonable to assume that the specular intensity IS' is proportional to $L(2\theta_n,\theta_s)$, while the constant of proportionality is dependent on the exact shape of the two components. To this end, we will use the constant B', rather than B, to represent the strength of the specular component of the photometric function.

The *modified photometric function* relates image intensity I' to extended source direction θ_s, and is expressed as the sum of the components IL' and IS':

$$I' = A' \cos(\theta_s - \theta_n) + B' L(2\theta_n, \theta_s) \,. \tag{9}$$

Since the parameters A' and B' are proportional to the parameters A and B, respectively, they may be used to represent the reflectance properties of the surface point.

3.3 Sampling

The process of measuring image intensities corresponding to different source directions is equivalent to sampling the modified photometric function $I'(\theta_s)$, as shown in Figure 6. Samples of the photometric function may be obtained by moving an extended source around the object and obtaining images of the object for different source positions. An alternative approach would be to distribute an array of extended sources around the object such that each source illuminates the object from a different direction. The entire array of extended sources may be sequentially scanned such that, for each scan, a single source is active and an image of the object surface is obtained. We have chosen to use this alternative approach in our experiments. We will confine the sampling process to two dimensions; the surface normal vector, viewing direction vector, and source direction vectors for all extended sources, are coplanar. The sequential scanning of extended sources positioned in the directions $\{\theta_i$: $i=1,2,......M\}$ results in a set of image intensities $\{I'_i$: $i=1,2,......M\}$ measured at each point on the object surface.

The number of samples measured at each surface point is determined by the frequency f at which $I'(\theta_s)$ is sampled. In order to extract the shape and the reflectance parameters of hybrid surfaces, both Lambertian and specular components of image intensity must be detected. Since we have used a delta function for the specular reflection model, the period of the modified photometric function that contains specular intensities is equal to the width, 2α, of the extended source radiance function. In the following section, we will show that, in general, at least two photometric samples must have non-zero specular intensities for the extraction technique to work. Hence, the photometric function must be sampled at a frequency greater than or equal to the *minimum sampling frequency*[3] $f_{\min}$, where:

$$f_{\min} = \frac{1}{\alpha} \,. \tag{10}$$

Note that, at this minimum frequency, the radiance distributions of adjacent extended sources overlap each other for an interval of α.

[3] It is assumed that the interval of the modified photometric function that contains specular intensities is small compared to the total width of the photometric function. Therefore, sampling frequencies that ensure the detection of specular intensities will provide a sufficient number of Lambertian intensity samples.

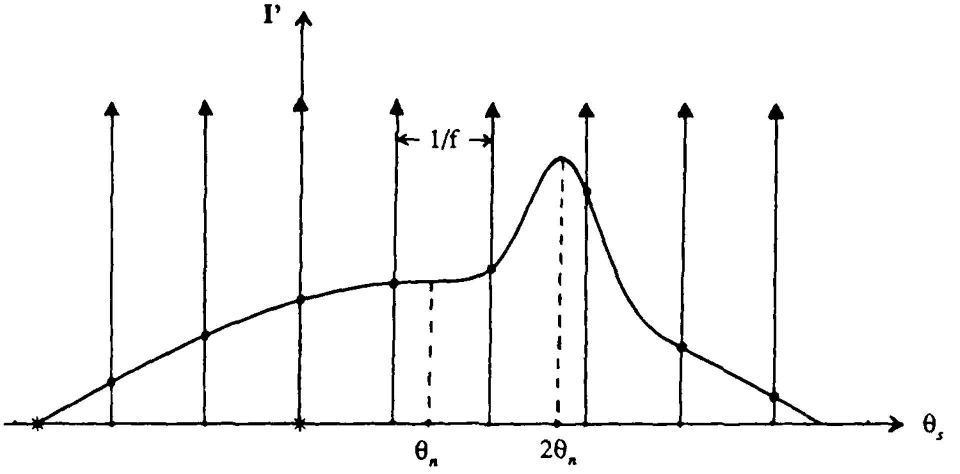


Figure 6: Sampling the modified photometric function.

4 Extracting Shape and Reflectance

Given the set of image intensities $\{I'_i\}$ measured at a surface point, we want to determine the orientation θ_n and reflectance parameters A' and B' of the point. We will first develop techniques to compute orientations of purely Lambertian and purely specular surfaces. Later, these techniques will be used to extract orientations and reflectance parameters of *all* instances of hybrid surfaces.

4.1 Lambertian Surfaces

Consider the case where the surface of an object is known to be purely Lambertian, and the shape of the object is to be determined. The photometric samples for a Lambertian surface point may be written as:

$$I'_i = A' \, cos(\theta_i - \theta_n) . \tag{11}$$

We would like to compute the orientation θ_n and A', the strength of the Lambertian reflection component. To this end, an error E is formulated as the sum of the errors in measured samples over the entire set of samples:

$$E = \sum_{i=1}^{M} [I'_i - A' \, cos(\theta_i - \theta_n)]^2 . \tag{12}$$

By using the conditions

$$\frac{\partial E}{\partial \theta_n} = 0 \quad \text{and} \quad \frac{\partial E}{\partial A'} = 0 , \tag{13}$$

we can determine values of θ_n and A' that minimize the error E.

4.2 Specular Surfaces

Now consider the case where the object surface is known to be purely specular, and the shape of the object is to be determined. The photometric samples for a specular point may be written as:

$$I'_i = B' L(2\theta_n, \theta_i) . \tag{14}$$

We want to determine the orientation θ_n and the specular strength B' from the intensity set $\{I'_i\}$. Let us assume that the specular direction $2\theta_n$ lies between the directions θ_k and θ_{k+1} of two adjacent extended sources. Further, let us assume that the photometric function is sampled using the minimum frequency f_{min}, i.e. $\theta_{k+1} = \theta_k + \alpha$. Then, since the surface is specular, only the samples I'_k and I'_{k+1} will have non-zero values. We see that when θ_n increases, $2\theta_n$ approaches θ_{k+1}, I'_k decreases, and I'_{k+1} increases. Similarly, when θ_n decreases, $2\theta_n$ approaches θ_k, I'_k increases, and I'_{k+1} decreases. In fact, from equation 14 we see that the intensity ratio I'_k/I'_{k+1} is equal to the ratio $L(2\theta_n, \theta_k)/L(2\theta_n, \theta_{k+1})$. Since the extended sources have decaying radiance functions (Appendix A), this ratio is a monotonic function of the angle $2\theta_n$. Since the radiance functions of the extended sources are known a-priori, we can precompute and store in memory the correspondence between I'_k/I'_{k+1} and θ_n.

Given the image intensity set $\{I'_i\}$ at a specular surface point, the non-zero image intensities in the set are first determined. If only a single intensity value, for instance I'_k, is greater than zero, then we know that $2\theta_n = \theta_k$. If two image intensities, for instance I'_k and I'_{k+1}, are greater than zero, the ratio I'_k/I'_{k+1} is used to determine θ_n. Once θ_n is found, B' is obtained by using equation 14:

$$B' = \frac{I'_k}{L(2\theta_n, \theta_k)} . \tag{15}$$

4.3 Hybrid Surfaces

The modified photometric function for hybrid surfaces is given by equation 9. At each surface point, we want to determine A', B', and orientation θ_n from the measured samples $\{I'_i$: i=1,2,......M$\}$ of the photometric function. To this end, we will develop an algorithm that attempts to separate the Lambertian and specular components of each measured image intensity and then computes surface orientations using the methods given above for Lambertian and specular surfaces.

The extraction algorithm is based on two constraints, namely, the *sampling frequency constraint* and the *unique orientation constraint*. By sampling the modified photometric function at the minimum sampling frequency f_{min}, we can ensure that only two consecutive image intensities in the intensity set $\{I'_i\}$ contain non-zero specular components. For each k in the interval 0 < k < M, I'_k and I'_{k+1} are hypothesized as being the two intensities that have specular components. All remaining intensities in the set $\{I'_i$: i=1,2,......M$\}$ must represent only Lambertian components of reflection. These intensities are used to compute the surface orientation θ_{nl} and the Lambertian strength A' (Section 4.1). The Lambertian components IL'_k and IL'_{k+1} are determined and used to separate the specular components IS'_k and IS'_{k+1} from I'_k and I'_{k+1}, respectively. The surface orientation θ_{ns} and specular strength B' are computed from IS'_k and IS'_{k+1} (Section 4.2).

Next, the physical constraint that each surface point has a unique orientation is exploited. An estimate θ_{nk} of the orientation is found as a weighted average of the orientations θ_{nl} and θ_{ns}. The weights are proportional to the strengths of the two components of reflection. We support this method of weight selection because the surface orientation that is computed from intensities resulting from the stronger of the two reflection components is less sensitive to image noises and is, therefore, more reliable. An orientation error e_k is found by comparing θ_{nk} with θ_{nl} and θ_{ns}. Using the above approach, orientation errors are computed for all k, where 0 < k < M. The orientation and reflectance parameters computed for the value of k that minimizes the orientation error are assigned to the surface point under consideration. This process is repeated for all points on the object surface.

It is important to note that the extraction algorithm is also capable of determining shape and reflectance of purely Lambertian and purely specular surfaces.

Extraction Algorithm

Step 1: Let k = 1 and e_0 equal a large positive number.

Step 2: Assume that image intensities I'_k and I'_{k+1} consist of specular components of reflection. All intensities I'_i, where $i \neq k$ and $i \neq k+1$, and the Lambertian model are used to compute the surface orientation θ_{nl} and Lambertian strength A'_k (Section 4.1).

Step 3: The specular components IS'_k and IS'_{k+1} are separated from the image intensities I'_k and I'_{k+1}:

$$\begin{aligned} IS'_k &= I'_k - A'_k \, cos(\theta_k - \theta_{nl}), \\ IS'_{k+1} &= I'_{k+1} - A'_k \, cos(\theta_{k+1} - \theta_{nl}). \end{aligned} \tag{16}$$

If $IS'_k < 0$ or $IS'_{k+1} < 0$, set k = k + 1 and go to step 2.

Step 4: The surface orientation θ_{ns} and the specular strength B'_k are determined by using specular intensities IS'_k and IS'_{k+1} and the specular model (Section 4.2).

Step 5: The best estimate of surface orientation, for the kth iteration, is determined as:

$$\theta_{nk} = \frac{A'_k \, \theta_{nl} + B'_k \, \theta_{ns}}{A'_k + B'_k}. \tag{17}$$

The orientation error e_k is determined as:

$$e_k = \frac{A'_k \, | \theta_{nl} - \theta_{nk} | + B'_k \, | \theta_{ns} - \theta_{nk} |}{A'_k + B'_k}. \tag{18}$$

Step 6: If $e_k < e_{k-1}$, then:

$$\theta_n = \theta_{nk}, \; A' = A'_k, \; B' = B'_k. \tag{19}$$

If k < M-1, set k = k + 1 and go to step 2. Otherwise, stop.

5 Experiments

5.1 Experimental Set-Up

We have conducted experiments to demonstrate the practical feasibility of the photometric sampling concept. A photograph of the experimental set-up used to implement photometric sampling is shown in Figure 7. A 14-inch diameter lamp shade is used as the spherical diffuser, and extended light sources are generated on the diffuser's surface by illuminating it using incandescent light bulbs. All light bulbs are assumed to have the same radiant intensity and are equidistant from the center of the diffuser. In our experiments, a source termination angle of $\alpha = 32$ degrees was used, and sampling was performed at the minimum frequency determined by equation 10. The object is placed at the center of the diffuser and is viewed by a camera through a 1-inch diameter hole at the top of the diffuser. The current set-up uses a WV-22 model Panasonic CCD camera that has a 512×480 pixel resolution. The complete imaging system, comprised of lenses and camera, has a physical resolution of 0.002 inches per pixel width. In the current implementation, the light bulbs, camera, and object are all placed in the same plane. This two-dimensional set-up is capable of measuring only the orientation of surface normal vectors that lie on a single plane in orientation space. For each extended source, an image of the object is digitized and stored in memory. The sequence of object images, generated by scanning the array of extended sources, is processed on a 3/60 SUN work-station.

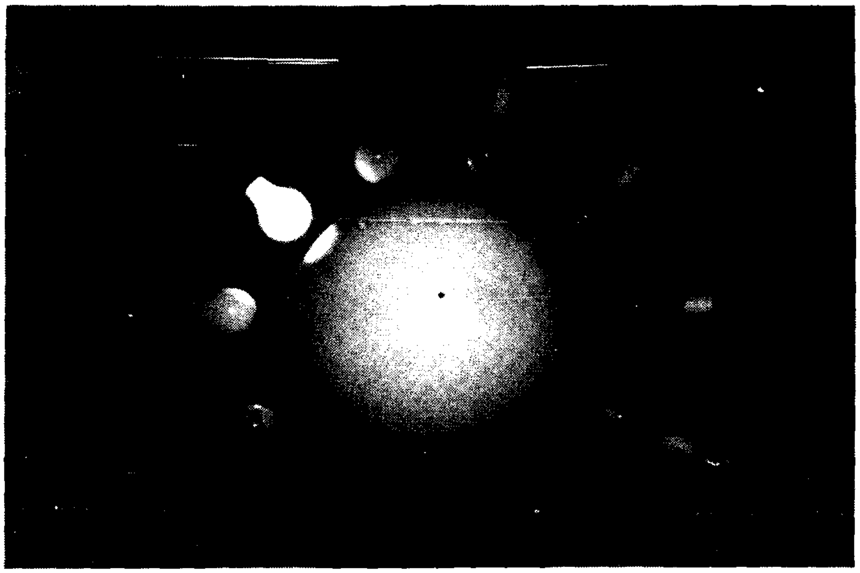


Figure 7: Photograph of the experimental set-up used to demonstrate the photometric sampling concept.

5.2 Experimental Results

Figure 8 shows photometric samples measured at a point on the surface of a plastic object using the above experimental set-up. The measured intensity values are represented by black dots. The reflectance model of the plastic surface includes both Lambertian and specular components. The orientation of the surface point was known a-priori. Using the orientation value, the two measured samples that were expected to consist of both Lambertian and specular intensities were identified and are marked in the figure as "L+S". All remaining image intensities result from Lambertian reflection and are marked in the figure as "L". The cosine function that best fits the Lambertian intensities is represented by the solid curve. The specular components were extracted from the two intensities that are marked as "L+S". Two estimates of surface orientation were computed using the Lambertian and the specular components of the image intensities. Both computed orientations were found to be within 2.5 degrees of the actual orientation value. Similar experiments were conducted on Lambertian and specular surfaces [10]. The results indicated that the reflectance model used in this paper does quite well in describing scattering of light by smooth surfaces.

The experimental set-up and the extraction algorithm were used to extract surface properties of a number of objects. Figures 9, 10, 11, and 12 show the results of the extraction method applied to objects with different surface reflectance properties. For each object, a photograph of the object is followed by two reflectance images and a needle map produced by the extraction algorithm, and a depth map that is constructed from the needle map. The reflectance properties of the surfaces are given by two images: the Lambertian strength image and the specular strength image. The intensity at each pixel, in both of these images, is proportional to the strength of the reflectance model component the image represents. The needle map is a representation of surface orientations. At each point on a needle map, the length of the needle is proportional to the tilt of the surface away from the viewing direction of the camera. The direction in which each needle points is determined by locating the starting point of the needle. All needles originate from the dots that constitute the resolution grid of the needle map. To help evaluate the performance of the extraction algorithm, we have included a depth map of each object that is obtained by integrating the orientations in the needle map. Note that the reconstructed surfaces are displayed at some arbitrary off-set level in all the depth maps.

The object shown in Figure 9 is cylindrical and its surface is Lambertian. Figure 10 is the photo of prism-shaped object that has a highly specular surface. An interesting application for the proposed method is seen in Figure 11. The object is a metal bolt that has a hexagonal-shaped head. The painted surface of the head is Lambertian in reflection, while the threaded section of the bolt is specular. Surface orientations are measured only along the thin edges of the threads since surface orientations in the grooves between threads lie outside the range of orientations that the current two-dimensional system is capable of measuring. While generating needle maps, surface orientations are sampled to make room for the display of needles. In the process of sampling, a considerable number of orientations measured on the threads of the bolt are lost. Hence, the orientations measured on a few threads are displayed at a higher resolution. We have not included a depth map of the bolt as its orientation map has many disjoint regions and is difficult to integrate.

All the above experiments were conducted on surface points that are either Lambertian or specular. A major advantage of the photometric sampling method, over other shape extraction techniques, lies in its ability to determine the shape and reflectance of hybrid surfaces. The surfaces of many manufactured plastic objects seem to fall into this category. The Lambertian component is produced by the internal scattering mechanism, while the sharp specular component results from the smoothness of the surface. Figure 12 shows the photo of a plastic object that is cylindrical in shape. As expected, non-zero Lambertian and specular strengths are seen in the reflection images. The needle and depth maps of the object are consistent with the actual shape.

An important feature of all the above results is that the surface properties at a pixel are computed solely from the intensities recorded at that pixel. The needle maps and reflectance images have not been subjected to any filtering operations. A simple error analysis was conducted to estimate the measurement accuracy of the current set-up. In the results obtained so far, measured surface orientations were found to be within 4 degrees of the actual orientation values, and an average error of 2 degrees in orientation was estimated.

6 Conclusions

We conclude this paper with the following remarks:

- The hybrid reflection model is obtained by studying various reflection mechanisms and by making assumptions regarding the microscopic surface shape.
- The photometric sampling method uses uniformly distributed source directions to obtain multiple photometric measurements at each surface point.
- Surface illumination, using extended light sources, makes it possible to capture both Lambertian and specular reflections in the image intensities.
- The extraction algorithm uses the photometric samples to determine the shape and reflectance parameters of hybrid surfaces, including Lambertian and specular surfaces. Objects comprised of combinations of the aforementioned surface classes can also be handled by the algorithm.
- The extraction algorithm is local in that the orientation and reflectance of a surface point are computed solely from image intensities recorded at that point.
- Accurate orientation estimates are obtained by using both Lambertian and specular components of the image intensities.

We are currently in the process of extending the theory and experimental set-up to three dimensions. We are also interested in using a more general reflection model that would broaden the spectrum of surfaces that the described shape extraction method can handle.

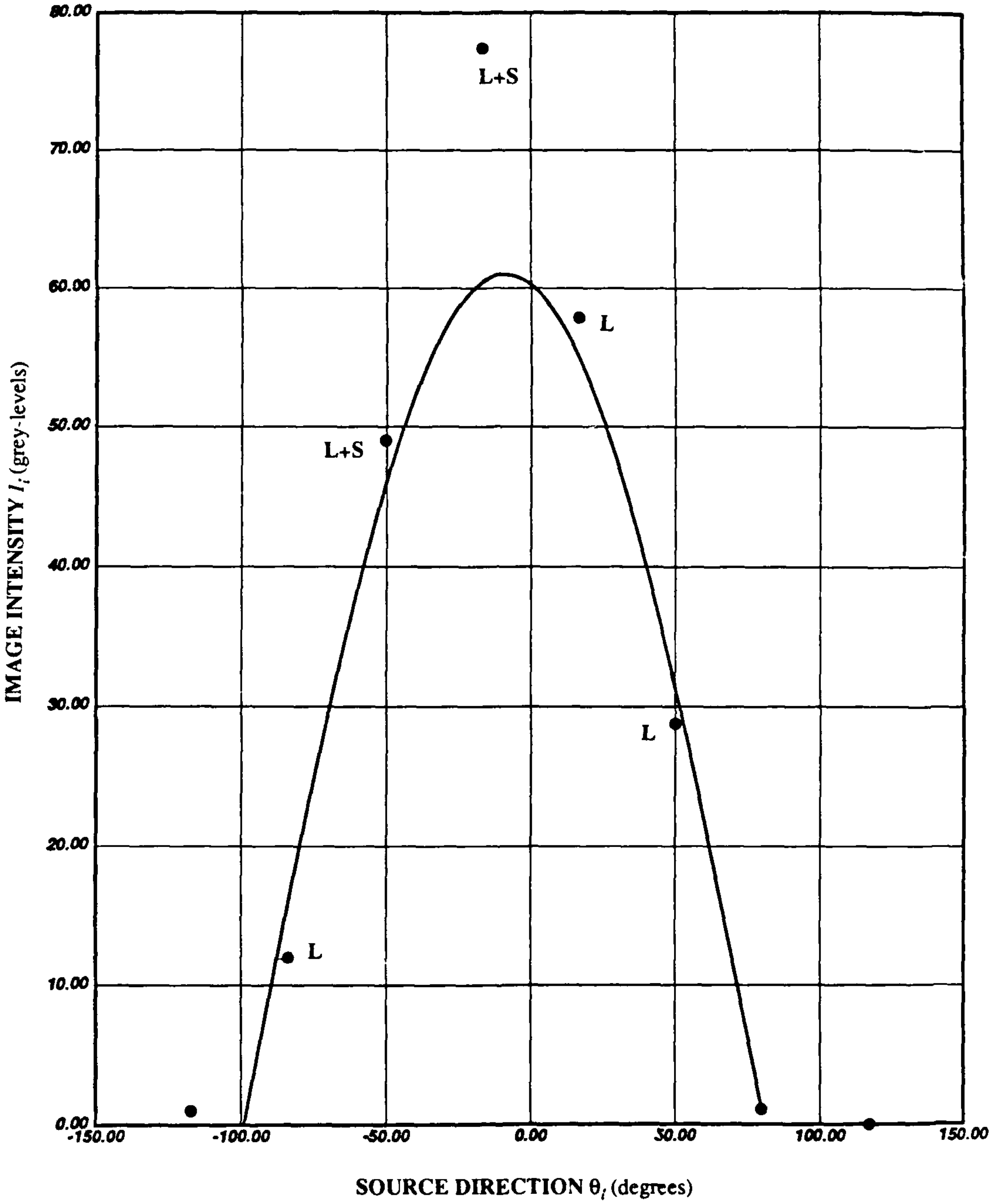


Figure 8: Samples of the photometric function, measured at a hybrid surface point. By using the known orientation of the surface point the two intensities that have specular reflections are identified and marked "L+S". The remaining points result solely from Lambertian reflection and the cosine function that best fits these points is shown as a solid curve.

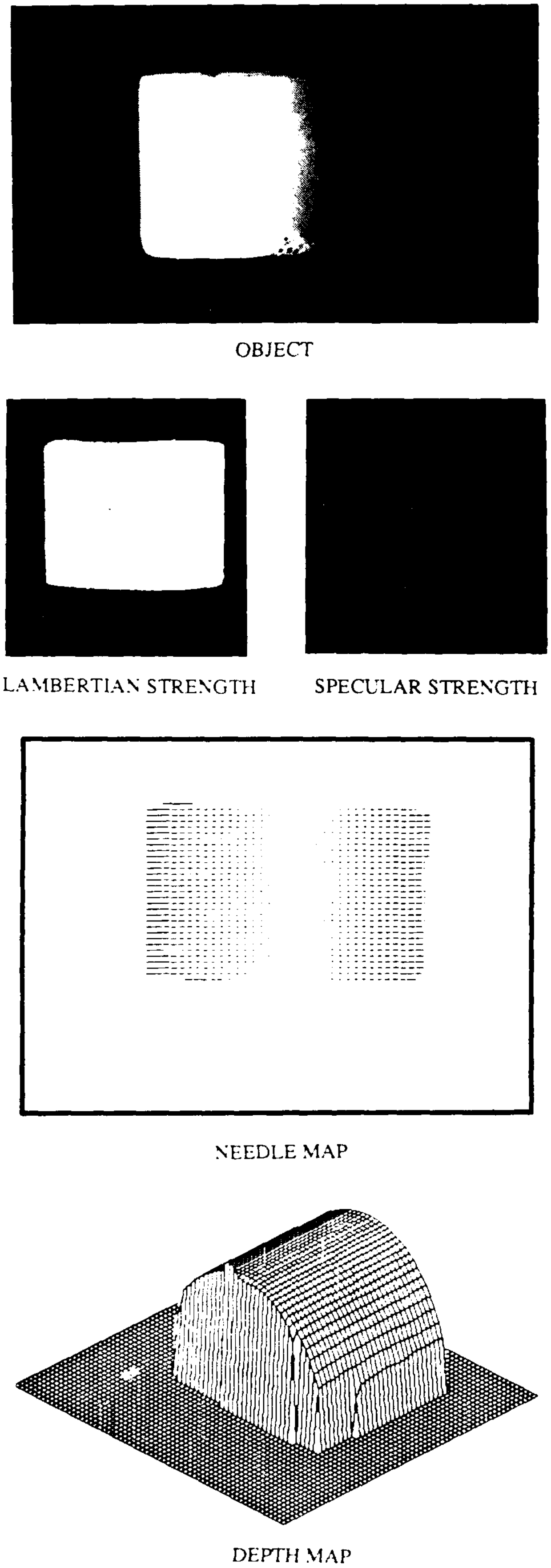


Figure 9: Cylindrical painted object with a Lambertian surface.

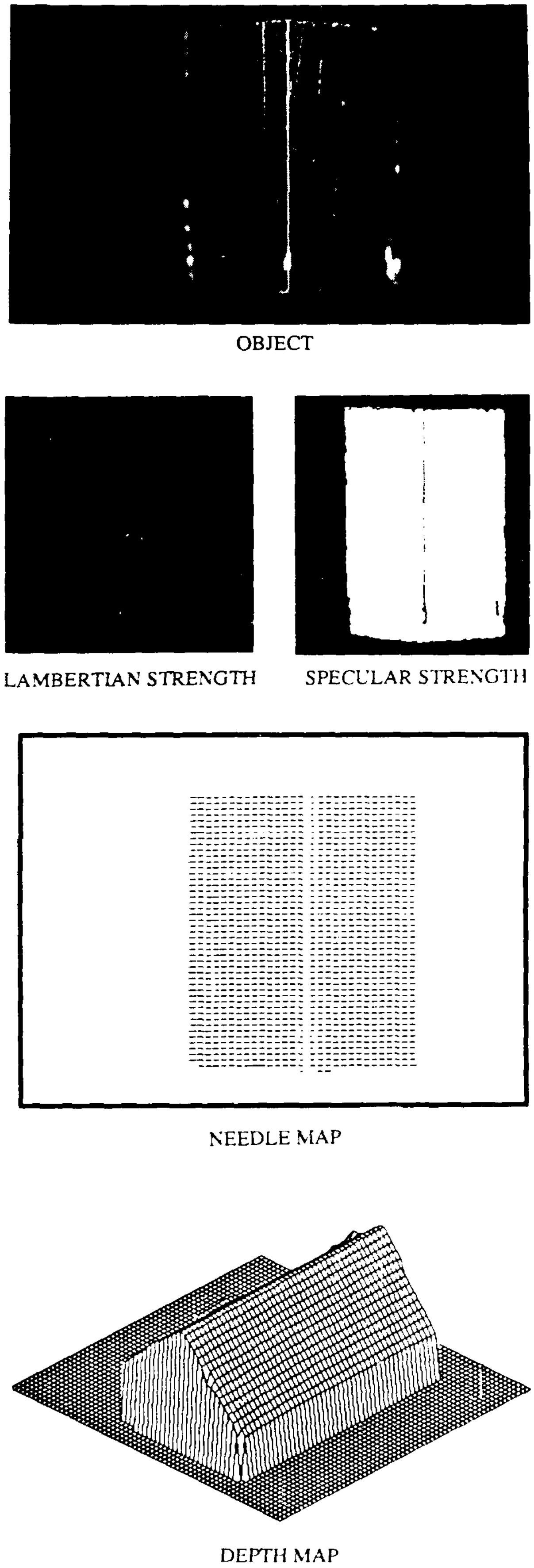
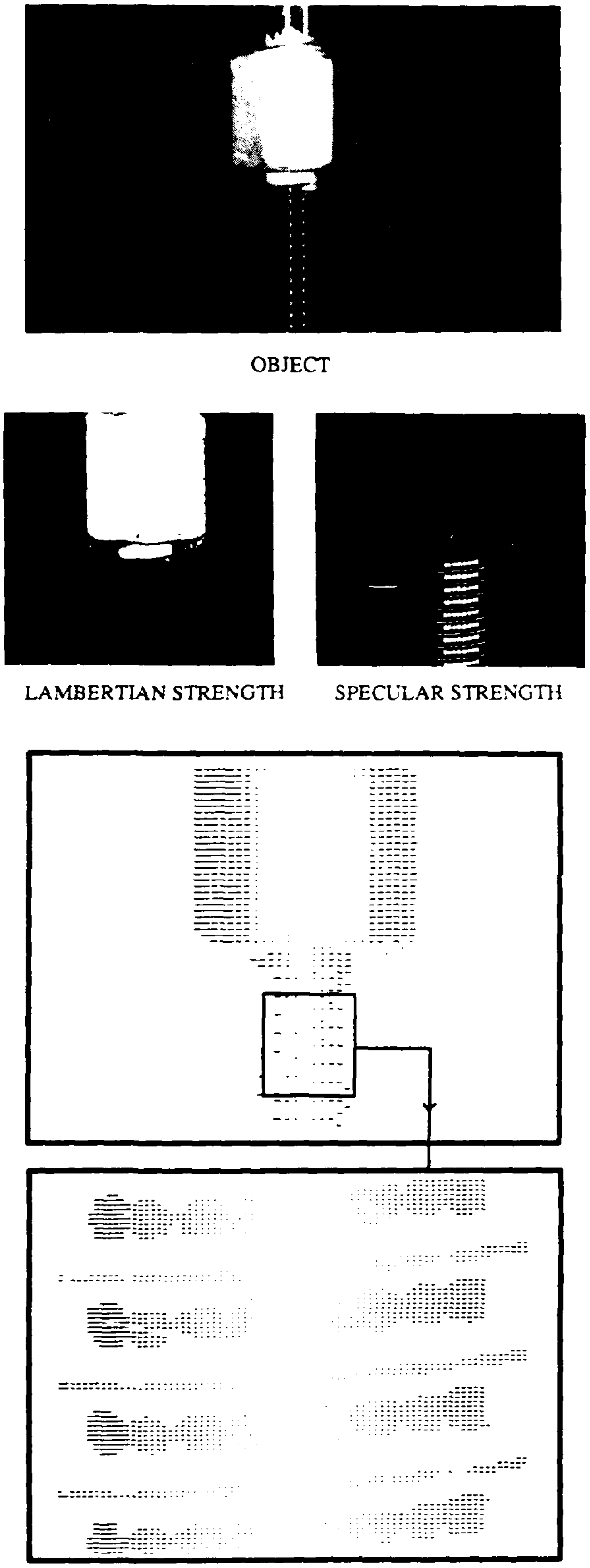


Figure 10: Prism-shaped metallic object with a specular surface.



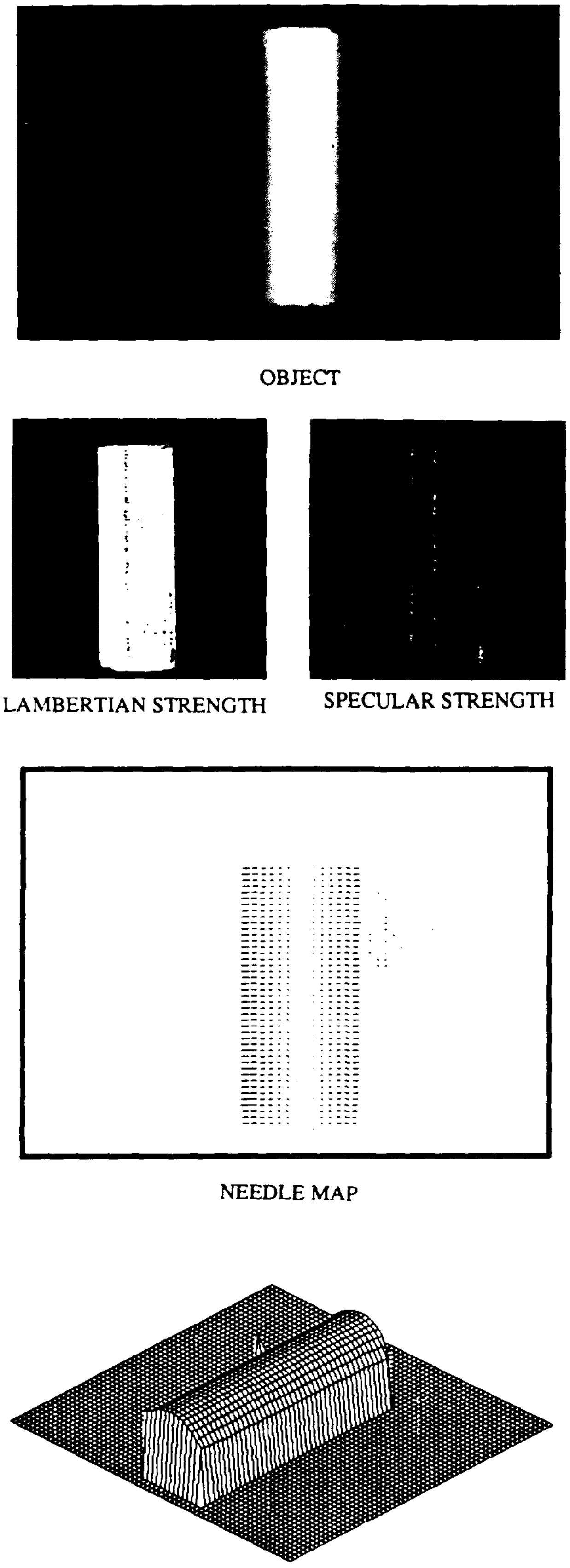
OBJECT

LAMBERTIAN STRENGTH

SPECULAR STRENGTH

NEEDLE MAP

Figure 11: A metal bolt. The head cf the bolt is painted and has a Lambertian surface, while the threaded section has a specular surface.



OBJECT

LAMBERTIAN STRENGTH

SPECULAR STRENGTH

NEEDLE MAP

DEPTH MAP

Figure 12: Cylindrical plastic object with a hybrid surface.

A Generating Extended Sources

There are numerous ways of generating extended light sources. In this section, we present the approach that we have chosen to use. An extended source can be generated by illuminating a sheet of light-diffusing material with a point light source. Figure 13 illustrates the illumination of a section of a circular diffuser of radius R. The point source is placed at a distance H from the diffuser's surface, and the viewed object is placed at the center of the circle. Let us assume that the diffuser is "ideal", i.e. incident energy is scattered equally in all directions. Then, the radiance[4] $L(\theta,\theta_s)$ of the inner surface of the diffuser is proportional to the irradiance[5] $E(\theta,\theta_s)$ of the outer surface of the diffuser:

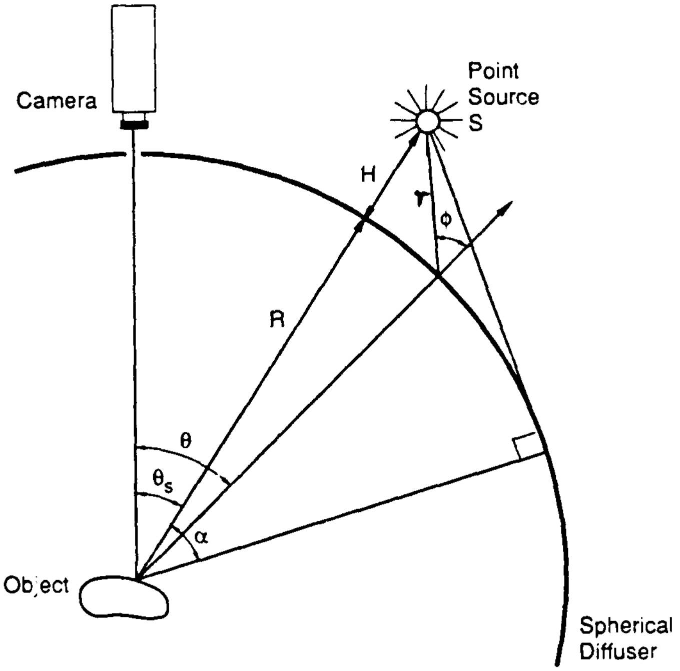


Figure 13: An extended source.

$$L(\theta,\theta_s) = CE(\theta,\theta_s)\,, \tag{20}$$

where C is a constant of proportionality. The analytic expression for the surface irradiance $E(\theta,\theta_s)$ may be derived from the basics of radiometry as:

$$E(\theta,\theta_s) = \frac{I\cos\phi}{r^2} \tag{21}$$

where I is the radiant intensity[6] of the point source S. The radiance of the extended source may be determined by expressing the variables r and ϕ in equation 21 in terms of the parameters R, H, and θ_s of the illumination

[4] Radiance is defined as the flux emitted per unit of foreshortened surface area per unit solid angle. Radiance is measured in watts per square meter per steradian ($W.m^{-2}.sr^{-1}$).

[5] Irradiance is defined as the incident flux density and is measured in watts per square meter ($W.m^{-2}$).

[6] Radiant Intensity of a source is defined as the flux exiting per unit solid angle and is measured in watts per steradian ($W.sr^{-1}$).

geometry:

$$L(\theta, \theta_s) = \frac{\mathrm{CI}[(R+H)cos(\theta - \theta_s) - R]}{[(R+H-Rcos(\theta - \theta_s))^2 + (Rsin(\theta - \theta_s))^2]^{3/2}} . \qquad (22)$$

Throughout this paper, the position of an extended source is denoted by the angle θ_s of the point source used to generate the extended source. The radiance function $L(\theta, \theta_s)$ is symmetric, or even, with respect to the source direction ($\theta = \theta_s$), and its magnitude decreases as θ deviates from θ_s. Points on the diffuser that lie in the interval θ_s-α < θ < θ_s+α receive light from the point source *S*. Points that lie outside this interval are occluded from the point source by points that lie in the interval. Thus, $L(\theta, \theta_s) = 0$ for $\theta < \theta_s$-α and $\theta > \theta_s$+α. The *source termination angle* α is determined from Figure 13 as:

$$\alpha = cos^{-1} \frac{R}{R+H} . \qquad (23)$$

Acknowledgements

The authors are grateful to B. K. P. Horn for his valuable comments. The members of the VASC center at Carnegie Mellon University provided many useful suggestions. Shree Nayar was supported by Westinghouse Electric Corporation.

References

[1] P. Beckmann and A. Spizzichino, *The Scattering of Electromagnetic Waves from Rough Surfaces,* The Macmillan Company, 1963.

[2] E. N. Coleman and R. Jain, *Obtaining 3-dimensional shape of textured and specular surface using four-source photometry,* Computer Graphics and Image Processing, Vol. 18, No. 4, pp. 309-328, April, 1982.

[3] G. Healey and T. O. Binford, *Local Shape from Specularity,* Proc. Image Understanding Workshop, Vol. 2, pp. 874-887, February, 1987.

[4] B. K. P. Horn, *Shape from Shading: A Methodfor Obtaining the Shape of a Smooth Opaque Object from One View,* MIT Project MAC Internal Report TR-79 and MIT AI Laboratory Technical Report 232, November, 1970.

[5] B. K. P. Horn, *Image intensity understanding,* Artificial Intelligence, Vol. 8, No. 2, 1977.

[6] K. Ikeuchi and B. K. P. Horn, *Numerical Shape from Shading and Occluding Boundaries,* Artificial Intelligence, Vol. 17, Nos. 1-3, pp. 141-184, August, 1981.

[7] K. Ikeuchi, *Determining surface orientations of specular surfaces by using the photometric stereo method,* IEEE Trans. on Pattern Analysis and Machine Intelligence, Vol. 3, No. 6, pp. 661-669, November, 1981.

[8] P. Kubelka and F. Munk, *Ein Beitrag sur Optik der Farbanstriche,* Z. tech. Physik, Vol. 12, 593, 1931.

[9] J. H. Lambert, *Photometria sive de mensura de gratibus luminis, colorum et umbrae,* Eberhard Klett, Augsburg, 1760.

[10] S. K. Nayar, K. Ikeuchi, T. Kanade, *Extracting Shape and Reflectance of Lambertian, Specular, and Hybrid Surfaces,* CMU-RI-TR-88-14, August, 1988.

[11] S. K. Nayar, K. Ikeuchi, T. Kanade, *Surface Reflection: Physical and Geometrical Perspectives,* CMU-RI-TR-89-7, March, 1989.

[12] F. E. Nicodemus, J. C. Richmond, J. J. Hsia, I. W. Ginsberg, and T. Limperis, *Geometrical Considerations and Nomenclature for Reflectance,* NBS Monograph 160, National Bureau of Standards, October 1977.

[13] A. P. Pentland, *Local Shading Analysis,* IEEE Trans. on Pattern Analysis and Machine Intelligence, Vol. 6, No. 2, pp. 170-187, March, 1984.

[14] J. Reichman, *Determination of Absorption and Scattering Coefficients for Nonhomogeneous Media. 1: Theory,* Applied Optics, Vol. 12, No. 8, pp. 1811-1815, August, 1973.

[15] A. C. Sanderson, L. E. Weiss, and S. K. Nayar, *Structured Highlight Inspection of specular surfaces,* IEEE Trans. on Pattern Analysis and Machine Intelligence, Vol. 10, No. 1, pp. 44-55, January, 1988.

[16] W. M. Silver, *Determining Shape and Reflectance Using Multiple Images,* S. M. Thesis, Dept. of Electrical Engineering and Computer Science, MIT, Cambridge, Massachusetts, June, 1980.

[17] K. Torrance and E. Sparrow, *Theory for Off-Specular Reflection from Roughened Surfaces,* Journal of the Optical Society of America, No. 57, pp. 1105-1114, 1967.

[18] R. J. Woodham, *Photometric stereo: A reflectance map technique for determining surface orientation from image intensity,* Proc. SPIE, Vol. 155, pp. 136-143, 1978.

Height and Gradient from Shading

Berthold K.P. Horn
Artificial Intelligence Laboratory
Massachusetts Institute of Technology

Abstract:

The shape-from-shading method described here enforces integrability and can deal with complex wrinkled surfaces. It allows fusion of shading information with height and gradient information obtained using other vision modalities such as binocular stereo or direct motion vision. The new method uses a regularizer to prevent divergence initially, but obtains the exact solution despite this, because it can drop the "departure from smoothness" penalty term once it gets near the solution. Two main features distinguish the new method from existing iterative schemes: simultaneous representation of both height and gradient and local linear expansion of the reflectance map about the present estimate of the gradient. If a number of implementation details are dealt with carefully, then the algorithm actually converges to the exact algebraic solution when presented with exact numerical data. Straightforward implementation leads to a scheme that, like other iterative schemes involving diffusion-like effects, takes time proportional to the square of the number of pixels in the image. This suggests that proper multi-grid implementation is called for.

The algorithm has been applied to synthetic images of smooth objects, as well as surfaces with discontinuities in gradient, such as crater-like shapes and polyhedra. The surface is recovered correctly given appropriate boundary conditions, without the need to first segment the image at the edges. Application to digital terrain models permits comparison of the recovered shape with the "ground truth."

1. Background

A special case of the shape-from-shading problem, applicable to surfaces with unusual reflecting properties, was solved by [Rindfleisch 66]. For the special reflectance properties he considered, a profile of the solution surface can be obtained by integrating along predetermined straight lines in the image plane. The general problem was formulated and solved in [Horn 70, 75]. There the method of characteristic strip expansion is used to solve the nonlinear first-order partial differential *image irradiance equation.* The *reflectance map* makes the analysis of shape-from-shading algorithm much easier, provided that both light sources and viewer are far away from the scene being viewed [Horn 77] [Horn & Sjoberg 79]. Several iterative schemes, mostly based on minimization of some functional containing an integral of the brightness error, arose later [Woodham 77] [Strat 79] [Ikeuchi & Horn 81] [Kirk 84] [Brooks & Horn 85] [Horn & Brooks 86] [Kirk 87] [Frankot & Chellappa 88]. For a collection of papers on shape from shading, as well as a review of the subject and an extensive bibliography, see *Shape from Shading* [Horn & Brooks 89].

The new method presented here was developed in part as a response to recent attention to the problem of integrability [Horn & Brooks 86] [Frankot & Chellappa 88], and exploits the idea of a coupled system of equations for depth and slope [Harris 86, 87] [Horn 88]. It borrows from well-known variational approaches to the problem [Ikeuchi & Horn 81] [Brooks & Horn 85], as well as an existing least-squares method for estimating surface shape given a needle map (see [Ikeuchi 84], chapter 11 in [Horn 86], and [Horn & Brooks 86]). For one choice of parameters, the new method becomes similar to one of the first iterative methods ever developed for shape from shading on a regular grid [Strat 79], while it degenerates into another well-known method [Ikeuchi & Horn 81] for a different choice of parameters. If, on the other hand, the brightness error term is dropped, then it degenerates into a well-known interpolation method [Harris 86, 87]. The computational effort grows rapidly with image size, so the new method can benefit from proper multigrid implementation [Brandt 77] [Brandt & Dinar 79] [Brandt 80, 84], as can existing iterative shape-from-shading schemes [Terzopolous 83, 84] [Kirk 84, 87].

It was found that a linear expansion of the reflectance map about the current estimate of the surface gradient leads to more rapid convergence. More importantly, this modification allows the scheme to converge in many cases where the simpler schemes diverge, or get stuck in local minima of the functional. Most existing iterative shape-from-shading methods handle only relatively simple surfaces. Such schemes can benefit from a retrofit of this idea.

The new scheme was tested on a number of synthetic images of increasing complexity, including some generated from digital terrain models of wrinkled surfaces, such as a glacial cirque with a number of gulleys. It recovered surface orientation at all points to within a degree or two in direction of the normal vector after a few hundred iterations. To attain this accuracy, several details of the implementation had to be carefully thought through [Horn 89]. Simpler surfaces are easier to process—with good results even when several of the implementation choices were not made in an optimal way. Similarly, these details may be less important for real images, were other error sources may dominate.

To conserve space, no detailed review of shape-from-shading work or photoclinometry is given here. For this the reader is referred to the collection of papers in [Horn & Brooks 89]. Similarly, there is not enough space to discuss several important implementation details—for these, see [Horn 89].

2. New Coupled Height and Gradient Scheme

The new shape-from-shading scheme will be presented through a series of increasingly more robust variational methods. We start with the simplest.

2.1 Fusing Height and Gradient Recovery

One way of fusing the recovery of gradient from shading with the recovery of height from gradient, is to represent both gradient (p, q) and height z in one variational scheme and to minimize something like

$$\iint \Big(\big(E(x,y) - R(p,q)\big)^2 + \mu\big((z_x - p)^2 + (z_y - q)^2\big) \Big)\, dx\, dy. \tag{1}$$

Note that, as far as $p(x, y)$ and $q(x, y)$, are concerned, this is an ordinary calculus problem (since no partial derivatives of p and q appear in the integrand). Differentiating the integrand with respect to $p(x, y)$ and $q(x, y)$ and setting the result equal to zero leads to

$$p = z_x + \frac{1}{\mu}(E - R)R_p \qquad \text{and} \qquad q = z_y + \frac{1}{\mu}(E - R)R_q. \tag{2}$$

Now $z(x, y)$ does not occur directly in $\big(E(x, y) - R(p, q)\big)$ so as far as height is concerned, we actually just need to minimize

$$\iint \big((z_x - p)^2 + (z_y - q)^2\big)\, dx\, dy. \tag{3}$$

The Euler equation for this variational problem [Horn 86] is just

$$\Delta z = p_x + q_y. \tag{4}$$

Altogether then we have one equation for each of the unknowns p, q and z.

These three equations are clearly satisfied when $p = z_x$, $q = z_y$ and $E = R$. That is, if a solution of the original shape-from-shading problem exists, then it satisfies this system of equations exactly (which is more than can be said for most other systems of equations for this problem obtained using a variational approach). It is instructive to substitute the expressions obtained for p and q (equation (2)) in $p_x + q_y$:

$$\begin{aligned} p_x + q_y = z_{xx} + z_{yy} + \frac{1}{\mu}\Big((E - R)\big(R_{pp}p_x + R_{pq}(p_y + q_x) + R_{qq}q_y\big) \quad & (5) \\ - \big(R_p^2 p_x + R_p R_q (p_y + q_x) + R_q^2 q_y\big) + \big(E_x R_p + E_y R_q\big) Bigr). & \end{aligned}$$

Since $\Delta z = (p_x + q_y)$ (equation (4)), we note that the three equations above for p, q and z are satisfied when

$$\big(R_p^2 p_x + R_p R_q (p_y + q_x) + R_q^2 q_y\big) - \big(E_x R_p + E_y R_q\big) = (E - R)\big(R_{pp}p_x + R_{pq}(p_y + q_x) + R_{qq}q_y\big). \tag{6}$$

This is exactly the equation obtained at the end of section 4.2 in [Horn & Brooks 86], where an attempt was made to directly impose integrability using the constraint $p_y = q_x$ (where it was stated that no convergent iterative scheme had been found for solving this complicated nonlinear partial differential equation directly).

Note that the natural boundary conditions for z are

$$c\, z_x + s\, z_y = c\, p + s\, q, \tag{7}$$

where (c, s) is a normal to the boundary.

The coupled system of equations above for p, q and z immediately suggests an iterative scheme

$$\begin{aligned} p_{kl}^{(n+1)} &= \{z_x\}_{kl}^{(n)} + \frac{1}{\mu}(E - R)R_p, \\ q_{kl}^{(n+1)} &= \{z_y\}_{kl}^{(n)} + \frac{1}{\mu}(E - R)R_q, \\ z_{kl}^{(n+1)} &= \bar{z}_{kl}^{(n)} + \frac{\epsilon^2}{\kappa}\left(\{p_x\}_{kl}^{(n+1)} + \{q_y\}_{kl}^{(n+1)}\right), \end{aligned} \tag{8}$$

where we have used the discrete approximation of the Laplacian for z:

$$\{\Delta z\} \approx \frac{\kappa}{\epsilon^2}(\bar{z}_{kl} - z_{kl}). \tag{9}$$

This new iterative scheme works well when the initial values given for p, q and z are close to the solution. In fact, it will converge to the exact solution if it exists, that is, if there exist a discrete set of values $\{z_{kl}\}$ such that $\{p_{kl}\}$ and $\{q_{kl}\}$ are the discrete estimate of the first partial derivatives of z with respect to x and y respectively and

$$E_{kl} = R(p_{kl}, q_{kl}) \tag{10}$$

In this case the functional we wished to minimize can actually be reduced to zero. It should be apparent that for this to happen, the estimator used for the Laplacian must match the sum of the convolution of the discrete estimator of the x derivative with itself and the convolution of the discrete estimator of the y derivative with itself[1].

The algorithm can easily be tested using synthetic height data z_{kl}. One merely estimates the partial derivatives using suitable discrete difference formulae and then uses the resulting values p_{kl} and q_{kl} to compute the synthetic image E_{kl}. This construction guarantees that there will be an exact solution. If a real image is used, there is no guarantee that there is an exact solution and the algorithm can at best find a good discrete approximation of the solution of the underlying continuous problem. In this case the functional will in fact not be reduced exactly to zero. In some cases the residue may be quite large. This may be the result of aliasing introduced when sampling the image, as discussed in [Horn 89], or because in fact the image given could not have arising from shading on a homogeneous surface with the reflectance properties and lighting as embodied in the reflectance map.

It turns out that the iterative algorithm developed in this section, while simple, is not very stable unless one is close to the exact solution, particularly when the surface is complex and the reflectance map not close to linear in the gradient. It can be improved greatly by linearizing the reflectance map. It can also be stabilized by adding a penalty term for departure from smoothness. This allows one to come close to the correct solution, at which point the penalty term is removed in order to prevent it from distorting the solution. Consider first the introduction of a penalty term for departure from smoothness.

2.2 Incorporating Departure from Smoothness Term

We now combine the iterative method of [Ikeuchi & Horn 81] for recovering p and q from $E(x,y)$ and $R(p,q)$ with the scheme for recovering z given p and q. We look directly for a minimum of

$$\iint \left(\left(E(x,y) - R(p,q)\right)^2 + \lambda(p_x^2 + p_y^2 + q_x^2 + q_y^2) + \mu\left((z_x - p)^2 + (z_y - q)^2\right) \right) dx\, dy. \tag{11}$$

The Euler equations of this calculus of variations problem lead to the following coupled system of second-order partial differential equations:

$$\begin{aligned} \lambda \Delta p &= -(E - R)R_p - \mu(z_x - p), \\ \lambda \Delta q &= -(E - R)R_q - \mu(z_y - q), \\ \Delta z &= p_x + q_y. \end{aligned} \tag{12}$$

[1] This and related matters are taken up in the implementation section of [Horn 89].

A discrete approximation of these equations can be obtained by using the discrete approximation of the Laplacian operator introduced above (equation (9)):

$$\{\Delta f\}_{kl} \approx \frac{\kappa}{\epsilon^2}(\overline{f}_{kl} - f_{kl}), \tag{13}$$

where $\overline{f}_{kl}$ is a local average of f_{kl}. Using the discrete approximation of the Laplacian we obtain:

$$\begin{aligned} \frac{\kappa\lambda}{\epsilon^2}(\overline{p}_{kl} - p_{kl}) &= -(E - R)R_p - \mu(z_x - p_{kl}), \\ \frac{\kappa\lambda}{\epsilon^2}(\overline{q}_{kl} - q_{kl}) &= -(E - R)R_q - \mu(z_y - q_{kl}), \\ \frac{\kappa}{\epsilon^2}(\overline{z}_{kl} - z_{kl}) &= p_x + q_y. \end{aligned} \tag{14}$$

where E, R, R_p, and R_q are the corresponding values at the point (k, l), while z_x, z_y, p_x and q_y are discrete estimates of the partial derivative of z, p and q there. We can collect all of the terms in p_{kl}, q_{kl} and z_{kl} on one side to obtain

$$\begin{aligned} (\kappa\lambda' + \mu)p_{kl} &= (\kappa\lambda'\overline{p}_{kl} + \mu z_x) + (E - R)R_p, \\ (\kappa\lambda' + \mu)q_{kl} &= (\kappa\lambda'\overline{q}_{kl} + \mu z_y) + (E - R)R_q, \\ \frac{\kappa}{\epsilon^2}\, z_{kl} &= \frac{\kappa}{\epsilon^2}\,\overline{z}_{kl} - (p_x + q_y), \end{aligned} \tag{15}$$

where $\lambda' = \lambda/\epsilon^2$. These equations immediately suggest an iterative scheme, where the right hand sides are computed using the current values of the z_{kl}, p_{kl}, and q_{kl}, with the results then used to supply new values for the unknowns appearing on the left hand sides[2].

From the above it may appear that $R(p, q)$, $R_p(p, q)$, and $R_q(p, q)$ should be evaluated using the "old" values of p and q. One might, on the other hand, argue that the local average values $\overline{p}$ and $\overline{q}$, or perhaps even the gradient estimates z_x and z_y, are more appropriate. Experimentation suggests that the scheme is most stable when the local averages $\overline{p}$ and $\overline{q}$ are used.

The above scheme contains a penalty term for departure from smoothness, that is, it has been regularized. Consequently it may appear that it cannot possibly converge to the exact solution, instead producing some smooth distorted surface. Indeed, it appears that the iterative scheme will "walk away" from the correct solution when it is presented with the solution as initial conditions, much as some earlier iterative schemes do. It turns out, however, that the penalty term is needed only to assure convergence when far from the solution. When we come closer to the solution, λ' can be reduced to zero and so the penalty term drops out. It is tempting to leave the penalty term out right from the start, since this simplifies the equations a great deal. The contribution from the penalty term does, however, help damp out instabilities when far from the solution and so is needed to avoid divergence in that situation.

2.3 Relationship to Existing Techniques

- Recently a new method has been developed that combines an existing iterative scheme for recovering surface orientation from shading with a projection onto the subspace of integrable gradients [Frankot & Chellappa 88]. Their approach is to alternately take one step of the iterative scheme [Ikeuchi & Horn 81] and then to find the integrable solution "nearest" to the result. The integrable gradient is then provided as initial conditions for the next step of the iterative scheme, thus ensuring that the gradient field never departs too far from integrability. The integrable gradient closest to a given gradient field is found using orthonormal series expansion and by exploiting the fact that differentiation in the spatial domain corresponds to multiplication by frequency in the transform domain.
- Similar results can be achieved by using instead the method described in [Ikeuchi 84] [Horn 86] [Horn & Brooks 86] for recovering the height $z(x, y)$ that best matches a given gradient. The resulting surface can

[2] These equations need to be solved iteratively both because the system of equations is so large and because of the fact that the reflectance map $R(p, q)$ is typically nonlinear.

then be differentiated to obtain initial values for $p(x,y)$ and $q(x,y)$ for the next step of the iterative scheme. This works, but not as well as the new scheme described in the previous section.

- Next, note that we obtain the scheme of [Ikeuchi & Horn 81] (who ignore the integrability problem) if we drop the departure from integrability term in the integrand—that is, when $\mu = 0$. If we instead remove the departure from smoothness term in the integrand—that is, when $\lambda = 0$—we obtain something reminiscent of the iterative scheme of [Strat 79], although Strat dealt with the integrability issue in a slightly different way.
- Finally, if we drop the brightness error term in the integrand, we obtain the scheme of [Harris 86, 87] for interpolating from depth and slope. He minimizes

$$\iint \left(\lambda(p_x^2 + p_y^2 + q_x^2 + q_y^2) + \left((z_x - p)^2 + (z_y - q)^2\right)\right) dx\, dy.$$

and arrives at the Euler equations

$$\lambda\,\Delta p = -(z_x - p), \quad \lambda\,\Delta q = -(z_y - q), \quad \text{and} \quad \Delta z = p_x + q_y. \tag{16}$$

Now consider that

$$\Delta(\Delta z) = \Delta(p_x + q_y). \tag{17}$$

Since application of the Laplacian operator and differentiation commute we have

$$\Delta(\Delta z) = (\Delta p)_x + (\Delta q)_y, \tag{18}$$

or

$$\lambda\,\Delta(\Delta z) = -(z_{xx} - p_x) - (z_{yy} - q_y), \tag{19}$$

and so

$$\lambda\,\Delta(\Delta z) = -\Delta z + (p_x + q_y) = 0. \tag{20}$$

So this method actually solves the bi-harmonic equation for z by solving a coupled set of Poisson's equations in an elegant, stable way that permits introduction of constraints on both height z and gradient (p,q). This is a good method for interpolating from sparse depth and surface orientation data.

The biharmonic equation has been employed to interpolate digital terrain models (DTMs) from contour maps. Such DTMs were used, for example, in [Horn & Bachman 78] [Horn 79] [Sjoberg & Horn 83]. The obvious implementations of finite difference approximations of the biharmonic operator, however, tend to be unstable because some of the weights are negative, and because the corresponding coefficient matrix lacks diagonal dominance. Also, the treatment of boundary conditions is complicated by the fact that the support of the biharmonic operator is so large. The scheme described above circumvents both of these difficulties.

2.4 Boundary Conditions & Nonlinearity of Reflectance Map

So far we have assumed that suitable boundary conditions are available, that is, the gradient is known on the boundary of the image region to which the computation is to be applied. If this is not the case, the solution is likely not to be unique. We may nevertheless find a solution by imposing so-called *natural* boundary conditions [Courant & Hilbert 62]. The natural boundary conditions for the variational problem described here can be shown to be

$$c\,p_x + s\,p_y = 0 \quad \text{and} \quad c\,q_x + s\,q_y = 0 \tag{21}$$

and

$$c\,z_x + s\,z_y = c\,p + s\,q \tag{22}$$

where (c,s) is a normal to the boundary. That is, the normal derivative of the gradient is zero and the normal derivative of the height has to match the slope in the normal direction computed from the gradient.

In the above we have approximated the original partial differential equations by a set of discrete equations, three for every picture cell (one each for p, q and z). If these equations were linear, we could directly apply all of the existing theory relating to convergence of various iterative schemes and how one solves such equations efficiently, given that the corresponding coefficient matrices are sparse[3]. Unfortunately, the equations are in

[3] See [Lee 88] for a discussion of the convergence of a particular iterative shape-from-shading scheme.

general not linear, because of the nonlinear dependence of the reflectance map $R(p,q)$ on the gradient components p and q. In fact, in deriving the above simple iterative scheme, we have essentially treated $R(p,q)$, and its derivatives, as constant (independent of p and q) during any particular iterative step.

2.5 Local Linear Approximation of Reflectance Map

We can do a lot better, while preserving the apparent linearity of the equations, by approximating the reflectance map $R(p,q)$ locally by a linear function of p and q. There are several options for choice of reference gradient for the series expansion, so let us keep it general for now at (p_0, q_0)[4]. We have

$$R(p,q) \approx R(p_0,q_0) + (p-p_0)\,R_p(p_0,q_0) + (q-q_0)\,R_q(p_0,q_0) + \cdots \tag{23}$$

Again, gathering all of the term in p_{kl} and q_{kl} on the left hand sides of the equations, we now obtain

$$\begin{aligned}(\lambda'' + R_p^2)\,p_{kl} + R_p R_q\,q_{kl} &= (\kappa\lambda'\overline{p}_{kl} + \mu z_x) + (E - R - p_0 R_p - q_0 R_q)R_p,\\ R_q R_p\,p_{kl} + (\lambda'' + R_q^2)\,q_{kl} &= (\kappa\lambda'\overline{q}_{kl} + \mu z_y) + (E - R - p_0 R_p - q_0 R_q)R_q,\end{aligned} \tag{24}$$

while the equation for z remains unchanged. Here we have abbreviated

$$\lambda'' = \kappa\lambda' + \mu. \tag{25}$$

It is convenient to rewrite these equations in terms of quantities relative to the reference gradient:

$$\begin{aligned}\delta p_{kl} &= p_{kl} - p_0 &\quad\text{and}\quad \delta q_{kl} &= q_{kl} - q_0\\ \delta\overline{p}_{kl} &= \overline{p}_{kl} - p_0 &\quad\text{and}\quad \delta\overline{q}_{kl} &= \overline{q}_{kl} - q_0\\ \delta z_x &= z_x - p_0 &\quad\text{and}\quad \delta z_y &= z_y - q_0\end{aligned} \tag{26}$$

This yields

$$\begin{aligned}(\lambda'' + R_p^2)\,\delta p_{kl} + R_p R_q\,\delta q_{kl} &= \kappa\lambda'\,\delta\overline{p}_{kl} + \mu\,\delta z_x + (E-R)R_p,\\ R_p R_q\,\delta q_{kl} + (\lambda'' + R_q^2)\,\delta q_{kl} &= \kappa\lambda'\,\delta\overline{q}_{kl} + \mu\,\delta z_y + (E-R)R_q.\end{aligned} \tag{27}$$

(The equations clearly simplify somewhat if we choose either $\overline{p}$ and $\overline{q}$ or z_x and z_y for the reference gradient p_0 and q_0.) We can view the above as a pair of linear equations for δp_{kl} and δq_{kl}. The determinant of the 2 × 2 coefficient matrix

$$D = \lambda''(\lambda'' + R_p^2 + R_q^2) \tag{28}$$

is always positive, so there is no problem with singularities. The solution is given by

$$\begin{aligned}D\,\delta p_{kl} &= (\lambda'' + R_q^2)\,A - R_p R_q\,B,\\ D\,\delta q_{kl} &= (\lambda'' + R_p^2)\,B - R_q R_p\,A,\end{aligned} \tag{29}$$

where

$$\begin{aligned}A &= \kappa\lambda'\,\delta\overline{p}_{kl} + \mu\,\delta z_x + (E-R)R_p,\\ B &= \kappa\lambda'\,\delta\overline{q}_{kl} + \mu\,\delta z_y + (E-R)R_q.\end{aligned} \tag{30}$$

(There are various interesting ways of rewriting these formulae). This leads to a convenient iterative scheme where the new values are given by

$$p_{kl}^{(n+1)} = p_0^{(n)} + \delta p_{kl}^{(n)} \quad\text{and}\quad q_{kl}^{(n+1)} = q_0^{(n)} + \delta q_{kl}^{(n)}, \tag{31}$$

in terms of the old reference gradient and the increments computed above. This new version of the iterative scheme does not require a great deal more computation, since the partial derivatives R_p and R_q are required in any case. It has been determined empirically that this scheme converges under a much wider set of circumstances than the one presented earlier.

Experimentation with different reference gradients, including the old values of p and q, the local average $\overline{p}$ and $\overline{q}$, as well as z_x and z_y showed that the accuracy of the solution as well as the convergence is affected by this choice. It became apparent that if we do not want the scheme to "walk away" from the correct solution, then we should use the old value of p and q for the reference p_0 and q_0.

[4] The reference gradient will, of course, be different at every picture cell, but to avoid having subscripts on the subscripts, we will simple denote the reference gradient at a particular picture cell by (p_0, q_0).

2.6 When to Stop Iterating

It is often difficult to decide when to stop an iteration. If we knew what the underlying surface was, we could just wait for the gradient of the solution to approach that of the surface. But, other than when we test the algorithm on synthetic images, we do not know what the surface is, otherwise we would probably not be using a shape-from-shading method in the first place! Some other tests include:

- The brightness error
$$\iint (E(x,y) - R(p,q))^2\, dx\, dy \tag{32}$$
should be small. Unfortunately this error becomes small after just a few iterations, so it does not yield a useful stopping criterion.
- The departure from smoothness
$$\iint (p_x^2 + p_y^2 + q_x^2 + q_y^2)\, dx\, dy \tag{33}$$
also drops as the solution is approached, but it does not constitute a particularly good indicator of approach to the solution. In particular, when one comes close to the solution, one may wish to reduce the parameters λ, perhaps even to zero, in which case further iterations may infact reduce smoothness in order to better satisfy the remaining criteria.
- One of the measures of lack of integrability
$$\iint ((z_x - p)^2 + (z_y - q)^2)\, dx\, dy \tag{34}$$
appears to be useful, since it drops slowly and often keeps on changing until the solution has converged.
- Another measure of lack of integrability
$$\iint (p_y - q_x)^2\, dx\, dy \tag{35}$$
is not quite as useful. since it can at times become quite small.or stop changing significantly, even when z is still inconsistent with p and q.
- One can also keep track of the rate of change of the solution with iterations
$$\iint \left(\frac{dp}{dt}\right)^2 + \left(\frac{dq}{dt}\right)^2 dx\, dy. \tag{36}$$
One should not stop until this has become quite small. In most cases it helps to continue for a while after the above measures stop changing rapidly; since the solution often continues to adjust.

3. Some Experimental Results

Shown in Figure 1 are synthetic images of a crater and of the shape recovered by the new algorithm. The image in Figure 1(a), corresponding to lighting from the Northwest, is the input provided to the algorithm, while Figure 1(c) is a synthetic image of the computed shape viewed under the same lighting conditions. Comparison of these two images, however, does not provide a useful test of the algorithm, since the brightness error, that is, the difference between these two images, becomes very small after just a few iterations, even though the surface shape at that stage of the computation is likely to still be quite inaccurate. To get an idea of how well such an algorithm really works, one needs to compare images of the original surface and that recovered by the algorithm under *different* lighting conditions. Shown in Figure 1(b) and 1(d) are images of the original surface and that recovered by the algorithm when the light source is placed in the Northeast. In this case the two images are identical, since the algorithm recovered the original shape with high accuracy after a few hundred iterations, under the assumption that the gradient is zero on the boundary of the image.

Many iterative schemes for shape from shading use the gradient to represent shape, and some of these schemes do not enforce integrability. In this situation showing an image of the recovered "surface" under the same lighting conditions as those used to obtain the input image is next to useless as a test of performance. In fact it is possible in this case to obtain a suitable gradient field in one step! That is, if neither integrability nor smoothness is enforced, the problem is so underconstrained that one can arrange for the "surface" to yield an arbitrary second image under different specified lighting conditions. Shown in Figure 3(a) and 3(b) are synthetic images of a digital terrain model for two different lighting conditions. In Figure 3(c) and 3(d) we see synthetic images of the computed gradient field "solution" under the same specified lighting conditions. The gradient field is computed by application of the photometric stereo method [Woodham 78, 80] at each point in the image[5]. This points out the importance of synthetic images of the recovered surface obtained with assumed lighting *different* from the lighting of the original scene.

To test the algorithm on more complex surfaces, a digital terrain model was interpolated from a portion of a contour map using methods developed earlier for work with digital terrain models [Horn & Bachman 78] [Horn 79] [Sjoberg & Horn 83]. Bradford Washburn of the Boston Museum of Science kindly supplied a new detailed contour map of the Mt. Washington region of the Presidential Range in the White Mountains of New Hampshire. The area chosen for this work is Huntington's ravine, a glacial cirque with several major gullies[6], a contour map of which is shown in Figure 2. Heights on a 113 × 85 grid were interpolated from the digitized contour map. Synthetic images of this surface illuminated from the Northwest and the Northeast are shown in Figure 4(a) and 4(b). The image shown in Figure 4(a) is provided as input to the algorithm. Given the surface gradient on the boundary, the algorithm converges to the exact solution (to machine precision) after several thousand iterations. Synthetic images of the computed surface after just a few hundred iterations are shown in Figure 4(c) and 4(d), under the same lighting conditions used for the images of the original surface.

4. Conclusion

A new iterative scheme for recovering shape from shading has been developed and implemented. The new scheme recovers height and gradient at the same time. Linearization of the reflectance map about the local average surface orientation greatly improves the performance of the new algorithm and could be used to improve the performance of existing iterative shape-from-shading algorithms. The new algorithm has been successfully applied to complex wrinkled surfaces.

5. Acknowledgements

Michael Caplinger kindly provided pointers to relevant publications on photoclinometry, in particular the Ph.D. thesis of R.L. Kirk [Kirk 87] at the California Institute of Technology. Michael Brooks drew my attention to the fact that existing iterative schemes "walk away" from the correct solution and that regularization does not "pick one of the infinite number of solutions" of an ill-posed problem, as sometimes mistakenly stated [Brooks 85]. Bradford Washburn supplied a detailed topographic map of the Mt. Washington area of the White Mountains in New Hampshire from which one of the digital terrain model used in the experiments here was interpolated. Joel Moses found a way for the Electrical Engineering and Computer Science Department to help cover the cost of the author's travel to the photoclinometry workshop held at Arizona State University, February 12-13, 1989, which provided much of the impetus for this paper.

6. References

Blake, A., A. Zisserman & G. Knowles (1985) "Surface Descriptions from Stereo and Shading," *Image & Vision Computing*, Vol. 3, No. 4, pp. 183-191. Also in (1989) *Shape from Shading*, Horn, B.K.P. & M.J. Brooks (eds.), MIT Press, Cambridge, MA.

Brandt, A. (1977) "Multi-level Adaptive Solutions to Boundary-Value Problems," *Mathematics of Computation*, Vol. 31, No. 138, April, pp. 333-390.

[5] This bizarre gradient field is, of course, not integrable and so does not correspond to any actual three-dimensional surface.

[6] The gullies are steep enough to be of interest to ice-climbers.

Brandt, A. (1980) "Stages in Developing Multigrid Solutions," in *Numerical Methods for Engineering*, Absi, E., R. Glowinski, P. Lascaux, H. Veysseyre (eds.), Dunod, Paris, pp. 23-44.

Brandt, A. (1984) *Multigrid Techniques: 1984 Guide with Applications to Fluid Dynamics*, Monograph available as GMD-Studie No. 85, from GMD-F1T, Postfach 1240, D-2505, St. Augustin 1, West Germany.

Brandt, A. & N. Dinar (1979) "Multigrid solutions of elliptic flow problems," in Parter, S.V. (ed.) *Numerical Methods for PDE*, Academic Press, New York, NY.

Brooks, M.J. (1985) Personal communication.

Brooks, M.J. & B.K.P. Horn (1985) "Shape and Source from Shading," *Proceedings of the International Joint Conference on Artificial Intelligence*, Los Angeles, CA, August 18-23, pp. 932-936. Also in (1989) *Shape from Shading*, Horn, B.K.P. & M.J. Brooks (eds.), MIT Press, Cambridge, MA.

Bruss, A.R. (1982) "The Eikonal Equation: Some Results Applicable to Computer Vision," *Journal of Mathematical Physics*, Vol. 23, No. 5, pp. 890-896, May. Also in (1989) *Shape from Shading*, Horn, B.K.P. & M.J. Brooks (eds.), MIT Press, Cambridge, MA.

Courant, R. & D. Hilbert (1962) *Methods of Mathematical Physics*, Volume I, Wiley, New York, NY.

Frankot, R.T. & R. Chellappa (1988) "A Method for Enforcing Integrability in Shape from Shading Algorithms," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol. 10, No. 4, pp. 439-451, July. Also in (1989) *Shape from Shading*, Horn, B.K.P. & M.J. Brooks (eds.), MIT Press, Cambridge, MA.

Harris, J.J. (1986) "The Coupled Depth/Slope Approach to Surface Reconstruction," S.M. Thesis, Department of Electrical Engineering and Computer Science, MIT. Also Technical Report 908, Artificial Intelligence Laboratory, MIT, Cambridge, MA.

Harris, J.J. (1987) "A New Approach to Surface Reconstruction: The Coupled Depth/Slope Model," *Proceedings of the International Conference on Computer Vision*, London, England, June 8-11, pp. 277-283.

Horn, B.K.P. (1970) "Shape from Shading: a Method for Obtaining the Shape of a Smooth Opaque Object from One View," Ph.D. Thesis, Department of Electrical Engineering, MIT. Also Technical Report TR-79, Project MAC, MIT, Cambridge, MA. Also Technical Report TR-232, Artificial Intelligence Laboratory, MIT, Cambridge, MA.

Horn, B.K.P. (1975) "Obtaining Shape from Shading Information," Chapter 4 in *The Psychology of Computer Vision*, P.H. Winston (ed.), McGraw Hill, New York, NY, pp. 115-155. Also in (1989) *Shape from Shading*, Horn, B.K.P. & M.J. Brooks (eds.), MIT Press, Cambridge, MA.

Horn, B.K.P. (1977) "Understanding Image Intensities (sic)," *Artificial Intelligence*, Vol. 8, No. 2, pp. 201-231, April. Also in (1987) *Readings in Computer Vision*, Fischler, M.A. & O. Firschein (eds.), Kaufmann, pp. 45-60.

Horn, B.K.P. (1979) "Automatic Hill-Shading and the Reflectance Map," *Image Understanding Workshop*, Palo Alto, CA, April 24-25, pp. 79-120. Also, *Proceedings of the IEEE*, Vol. 69, No. 1, pp. 14-47, January. Also (1982) *Geo-Processing*, Vol. 2, No. 1, pp. 65-146, October.

Horn, B.K.P. (1986) *Robot Vision*, MIT Press, Cambridge, MA & McGraw-Hill, New York, NY.

Horn, B.K.P. (1988) "Some Ideas on Parallel Analog Networks for Machine Vision," Memo 1071, Artificial Intelligence Laboratory, MIT, Cambridge, MA, December.

Horn, B.K.P. (1989) "Height and Gradient from Shading," Memo 1105, Artificial Intelligence Laboratory, MIT, Cambridge, MA, March.

Horn, B.K.P. & B.L. Bachman (1978) "Using Synthetic Images to Register Real Images with Surface Models," *Communications of the ACM*, Vol. 21, No. 11, pp. 914-924, November. Also "Registering Real Images Using Synthetic Images," in *Artificial Intelligence: An MIT Perspective* (Volume II), Winston, P.H. & R.H. Brown (eds.), MIT Press, Cambridge, MA, pp. 129-160.

Horn, B.K.P. & M.J. Brooks (1986) "The Variational Approach to Shape from Shading," *Computer Vision, Graphics and Image Processing*, Vol. 33, No. 2, pp. 174-208, February. Also in (1989) *Shape from Shading*, Horn, B.K.P. & M.J. Brooks (eds.), MIT Press, Cambridge, MA.

Horn, B.K.P. & M.J. Brooks (eds.) (1989) *Shape from Shading*, MIT Press, Cambridge, MA, May.

Horn, B.K.P. & R.W. Sjoberg (1979) "Calculating the Reflectance Map," *Applied Optics*, Vol. 18, No. 11, pp. 1770-1779, June. Also in (1989) *Shape from Shading*, Horn, B.K.P. & M.J. Brooks (eds.), MIT Press, Cambridge, MA.

Ikeuchi, K. (1984) "Reconstructing a Depth Map from Intensity Maps," *International Conference on Pattern Recognition*, Montreal, Canada, July 30–August 2, pp. 736–738. Also (1983) "Constructing a Depth Map from Images," Memo 744, Artificial Intelligence Laboratory, MIT, Cambridge, MA, August.

Ikeuchi, K. & B.K.P. Horn (1981) "Numerical Shape from Shading and Occluding Boundaries," *Artificial Intelligence*, Vol. 17, No. 1–3, pp. 141–184, August. Also in (1989) *Shape from Shading*, Horn, B.K.P. & M.J. Brooks (eds.), MIT Press, Cambridge, MA.

Kirk, R.L. (1984) "A Finite-Element Approach to Two-Dimensional Photoclinometry," brief abstract in *Bulletin of the American Astronomical Society*, Vol. 16, No. 3, pg. 709.

Kirk, R.L. (1987) "A Fast Finite-Element Algorithm for Two-Dimensional Photoclinometry," Part III of Ph.D. Thesis, Division of Geological and Planetary Sciences, California Institute of Technology, Pasadena, CA.

Lee, C.-H. & A. Rosenfeld (1985) "Improved Methods of Estimating Shape from Shading using the Light Source Coordinate System," *Artificial Intelligence*, Vol. 26, No. 2, pp. 125–143. Also in (1989) *Shape from Shading*, Horn, B.K.P. & M.J. Brooks (eds.), MIT Press, Cambridge, MA.

Lee, D. (1988) "Algorithms for Shape from Shading and Occluding Boundaries," *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, June 5-9, Ann Arbor, MI, pp. 478–485. Also in (1989) *Shape from Shading*, Horn, B.K.P. & M.J. Brooks (eds.), MIT Press, Cambridge, MA.

Malik, J. & D. Maydan (1989) "Recovering Three Dimensional Shape from a Single Image of Curved Objects," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol. 11, No. 4. Also in (1989) *Shape from Shading*, Horn, B.K.P. & M.J. Brooks (eds.), MIT Press, Cambridge, MA.

Pentland, A.P. (1984) "Local Shading Analysis," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol. 6, No. 2, pp. 170–187, March. Also in (1989) *Shape from Shading*, Horn, B.K.P. & M.J. Brooks (eds.), MIT Press, Cambridge, MA.

Pentland, A.P. (1988) "Shape Information from Shading: A Theory about Human Perception," Technical Report 103, Vision Sciences, MIT Media Laboratory, MIT, Cambridge, MA, May.

Rindfleisch, T. (1966) "Photometric Method for Lunar Topography," *Photogrammetric Engineering*, Vol. 32, No. 2, pp. 262–277, March. Also (1965) "A Photometric Method for Deriving Lunar Topographic Information," Technical Report 32-786, Jet Propulsion Laboratory, California Institute of Technology, Pasadena, CA, September.

Saxberg, B.V.H. (1988) "A Modern Differential Geometric Approach to Shape from Shading," Ph.D. Thesis, Department of Electrical Engineering and Computer Science, MIT, Cambridge, MA.

Sjoberg, R.W. & B.K.P. Horn (1983) "Atmospheric Effects in Satellite Imaging of Mountainous Terrain," *Applied Optics*, Vol. 22, No. 11, pp. 1702–1716, June.

Strat, T. (1979) "A Numerical Method for Shape from Shading for a Single Image," S.M. Thesis, Department of Electrical Engineering and Computer Science, MIT, Cambridge, MA.

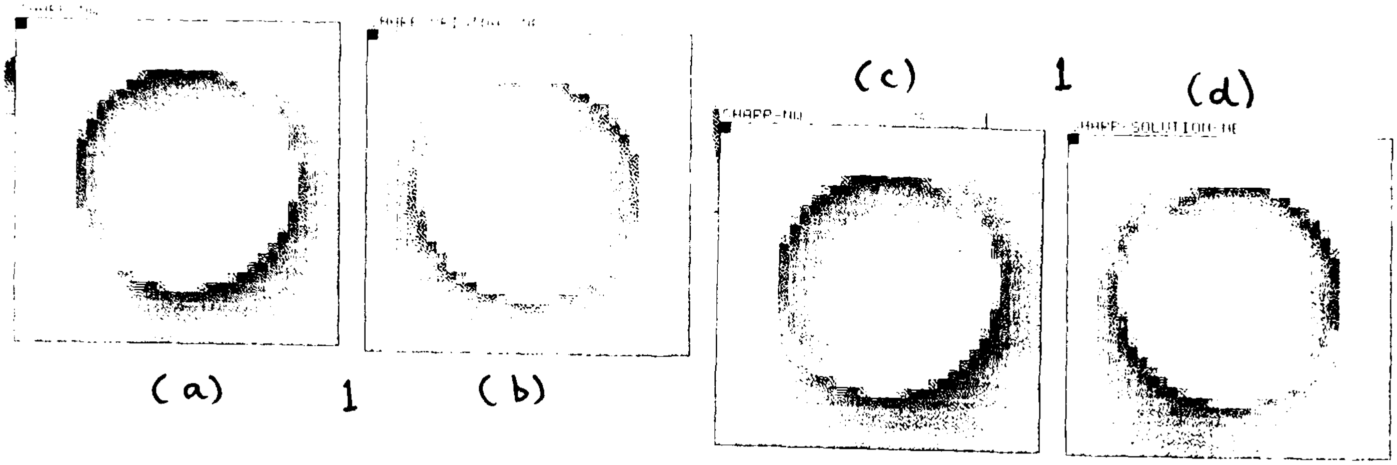
Terzopoulos, D. (1983) "Multilevel Computational Processes for Visual Surface Reconstruction," *Computer Vision, Graphics and Image Processing*, Vol. 24, pp. 52–96. Also (1982) "Multi-level Reconstruction of Visual Surfaces: Variational Principles and Finite Element Representation, Memo 671, Artificial Intelligence Laboratory, MIT, Cambridge, MA, April.

Terzopoulos, D. (1984) "Multigrid Relaxation Methods and the Analysis of Lightness, Shading, and Flow," Memo 803, Artificial Intelligence Laboratory, MIT, Cambridge, MA, October. Also chapter 10 in *Image Understanding 84*, Ullman, S. & W. Richards (eds.), Ablex Publishing Corporation, Norwood, NJ, pp. 225–262.

Woodham, R.J. (1977) "A Cooperative Algorithm for Determining Surface Orientation from a Single View," *International Joint Conference on Artificial Intelligence*, Cambridge, MA, August 22–25, pp. 635–641.

Woodham, R.J. (1978) "Photometric Stereo: A Reflectance Map Technique for Determining Surface Orientation from a Single View," *Image Understanding Systems & Industrial Applications, Proceedings of the Society of Photo-Optical Instrumentation Engineers*, Vol. 155, pp. 136–143.

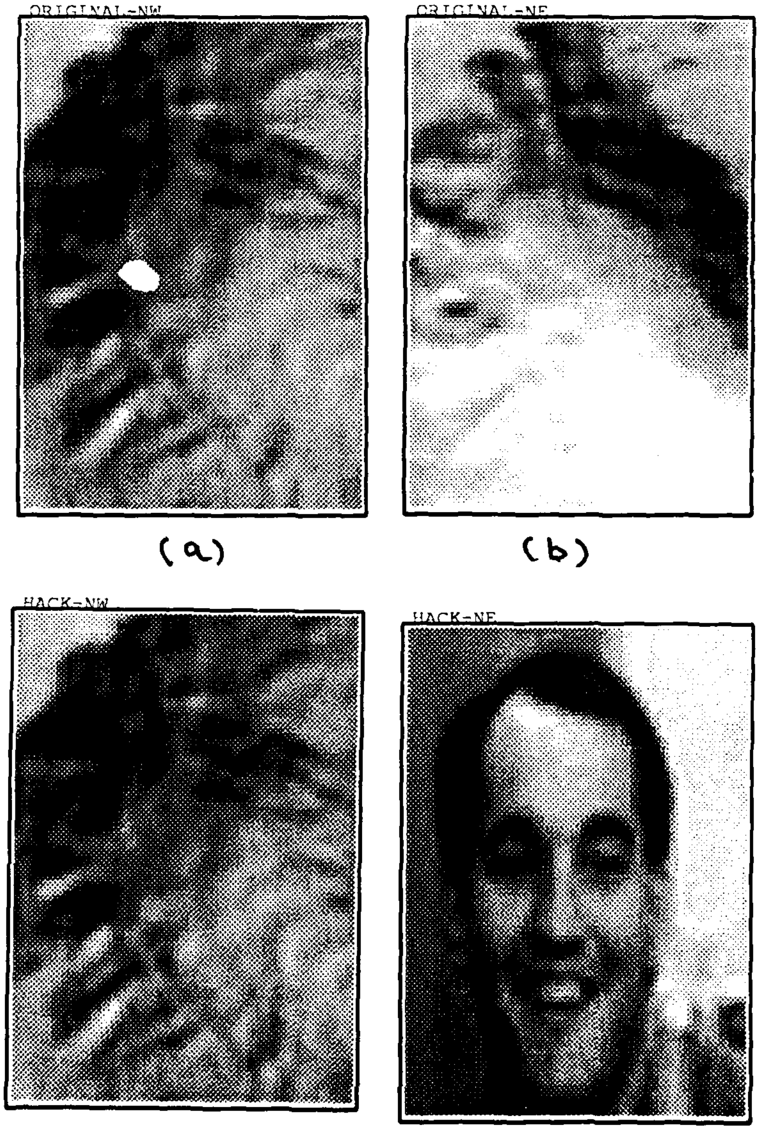
Woodham, R.J. (1980) "Photometric Method for Determining Surface Orientation from Multiple Images," *Optical Engineering*, Vol. 19, No. 1, January-February, pp. 139–144. Also in (1989) *Shape from Shading*, Horn, B.K.P. & M.J. Brooks (eds.), MIT Press, Cambridge, MA.



(a) 1 (b)

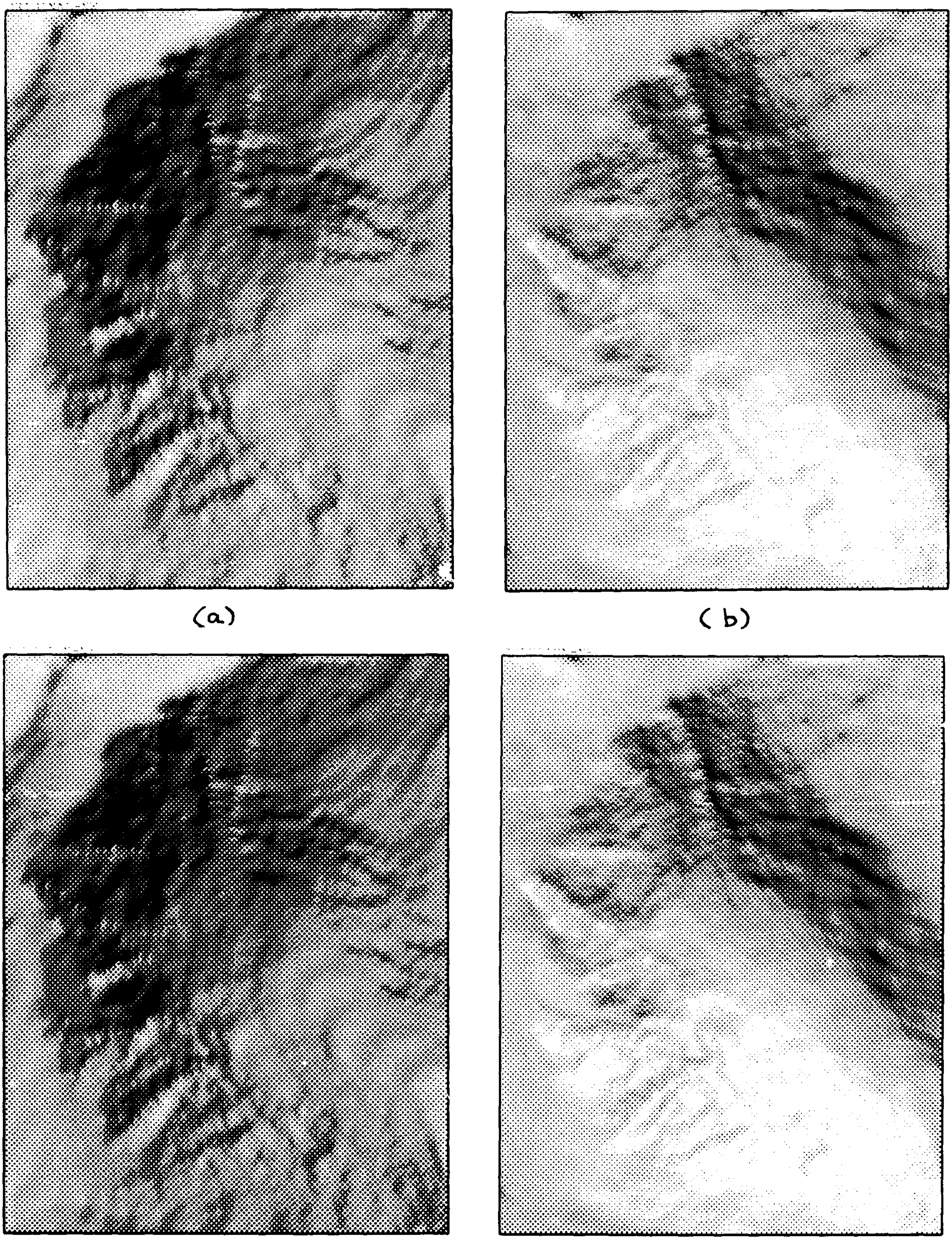
(c) 1 (d)

2



(a) (b)

(c) 3 (d)



(a)

(b)

(c)

(d)

4

SECTION III

OTHER TECHNICAL REPORTS

THF SECOND DARPA IMAGE UNDERSTANDING BENCHMARK ON WARP AND EXTENDING APPLY TO INCLUDE GLOBAL OPERATIONS

Jon A. Webb and Mike B. MacPherson
School of Computer Science
Carnegie-Mellon University
Pittsburgh, PA 15213

ABSTRACT

Warp was a participant in the second DARPA Image Understanding Benchmark study, which compared the performance of a variety of architectures on a complete, integrated, image processing task. The performance and implementation of the benchmark on the Warp machine are presented. Both tightly-coupled data parallelism and loosely-coupled task-parallelism were exploited in order to get the best overall execution time on the task.

Using the experience on this task, programming issues for low- and mid-level vision on parallel computers are examined. In particular, the problem of extending the Apply language (a machine-independent language for low-level computer vision) is considered. Extensions to Apply are proposed and it is shown how these extensions allow the efficient implementation of all of the low- and mid-level vision operations in the benchmark. The class of local and global operations that can be programmed with the extended Apply is considered. It is shown that all local and global operations that are *reversible* – which produce the same results when applied from the top down, or from the bottom up – can be programmed in the extended Apply.

INTRODUCTION

The second DARPA Image Understanding Benchmark addressed the issue of system performance on an integrated set of tasks that interact in a way typical of complex vision applications. The goal of the benchmark study was to gain a better understanding of vision architecture requirements that can be used to guide the development of future vision architectures.

In the first DARPA Image Understanding Benchmark (Rosenfeld, 1987) the problems to be solved were specified, but the algorithms were not given. This led to a great variety of different approaches to implementing each of the algorithms, which made it difficult to compare different architectures. The performance of an architecture depended on the algorithm chosen and the degree of optimization in its implementation as well as the performance of the computer. In this study both the problem to be solved and the algorithm were specified, and C code was given to solve the problem on a Unix computer.

In this benchmark, a complete image recognition system – the recognition of a two-dimensional mobile – was tested. This is in contrast to the first benchmark, where individual routines were tested. This approach has the advantage that overall timing is more realistic, since it measures the times that will be encountered when actually using the architecture to do recognition.

In this paper, we present the results of our implementation of the benchmark on Warp, then turn to the issue of programming low- and mid-level vision in an machine-independent manner. We have already developed (and used in this study) a machine-independent language for local low-level image processing operations, called Apply. Use of this language in the benchmark led to fast implementations of some of the routines, compared to implementation of other routines with conventional tools. It would therefore be useful to be able to implement more of the routines with an Apply-like tool.

We propose specific extensions to Apply, then show how those extensions make it possible to program all of the low- and mid-level algorithms in this benchmark. We then consider what class of image processing operations can be programmed in the extended Apply. We will prove that it is the class of image processing operations that are *reversible*; that is, they produce the same result when applied to the image from the top down, or the bottom up.

BENCHMARK DESCRIPTION

The benchmark involves the recognition of an approximately specified 2 1/2 D "mobile" sculpture composed of rectangles, given images from intensity and range sensors.

A complete description of the benchmark is available elsewhere (Weems, et al., 1988). We describe it here in outline form.

The problem is to recognize a two-dimensional mobile given a range and intensity image. The mobile is shown in Figure 1. It consists of a number of rectangular regions, connected by invisible threads, oriented randomly in planes normal to the line of sight. The rectangular regions have different solid graylevels.

The input to the program is two 512×512 images. The first is a byte intensity image, the second is a floating-point depth image. The two images are registered, and are taken with parallel projection.

The program is given a set of 10 mobile models, and must choose the correct matching model from its set. There is only one correct match for each image. There are three factors that complicate recognition: the rectangles are allowed to rotate in the image plane around the invisible threads connecting them to the rest of the mobile, there are superfluous rectangles in the scene, and the depth image has added noise.

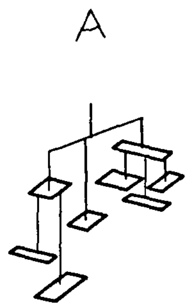


Figure 1: A Simple Mobile

The processing in the task is required to proceed in the following steps:

1. **Connected component labelling** on the input intensity image. Since it is noise-free, it does not have to be filtered or thresholded first. Connected component labelling will correctly identify the rectangles in the image as long as they do not partially cover spurious rectangles of the same color, or are hidden by other model or spurious rectangles.
2. **Trace** and **compute k-curvature** of the borders in the labelled intensity image.
3. **Smooth** and perform **zero-crossing** detection in the **first derivative** of the border k-curvature. This detects corners in the intensity image.
4. **Count corners** of components. If there are three right angles in sequence, declare a rectangle.
5. **Median filter** the depth image. Either 3×3 and 5×5 operators could be used.
6. Perform a **Sobel** operator on the smoothed depth image to detect edges.
7. Using a depth-first graph matching technique, match the hypothesized rectangles against the model.

Verify matches by **probing** the images, using two techniques:

a. **Depth** probe: within the hypothesized rectangle region, examine pixels; pixels deeper than the hypothesized rectangle count against it; pixels closer have no effect (since there could be a rectangle covering the hypothesized rectangle); and pixels at the right depth and of the ri¸ht grayvalue count for the rectangle.

b. **Hough** probe: perform a hough transform on the region of the processed depth image within the rectangle and look for peaks at a predicted location.

8. **Paint** the detected model over the intensity image and output the result.

This recognition procedure is not necessarily the fastest method for solving the problem. In fact, as was pointed out by C. H. Chien of Carnegie-Mellon, it is possible "short-circuit" the benchmark. A simple histogramming operation applied to the depth image is sufficient to obtain the depth information of most of the rectangles, and these rectangles can be extracted using a multi-value thresholding technique in a later stage. There may be cases where more than one rectangle has the same depth due to the resolution used in extracting depth information from the histogram of the depth image. These cases are rare and thus will not affect the result of matching.

The matching can be carried out in two steps: the initial matching and verification. The initial matching is trivial due to the assumption that all the rectangles are parallel to the image plane. This assumption allows us to impose an ordering on the rectangles in the mobile and to use "relative depth" as the *sole* feature for the initial matching. Note that the sizes, colors, positions and orientations of the rectangle have not been used up to this stage.

At the end of the initial matching, the depth of each rectangle of the mobile in the depth image can be easily determined, and this information can be used to guide the extraction of each rectangle in the depth image. As mentioned earlier a simple thresholding technique is enough to determine which depth pixels are associated with each rectangle. The corners of that rectangle will be the ones with extreme values in x- or y- coordinates. Three corners can uniquely determine a rectangle (a small number of additional depth pixels may be required in the presence of noise). Edge detection, boundary following, or connected component labeling are unnecessary for extracting the rectangle. The extracted information is then used to verify the result of the initial matching.

Notice that the intensity image has never been used at all. It may become necessary to resolve some ambiguity in the depth image by using the intensity image, but this can easily be done by probing the intensity image at certain points, without needing to go through any sort of elaborate processing.

We now review the Warp, then discuss each of the routines in turn.

WARP AND WARP PROGRAMMING

We briefly review the Warp hardware; further detail is available elsewhere (Annaratone, et al., 1987). Warp is a medium grain parallel computer. It consists of a linear pipeline of ten powerful cells, each with 10 MFLOPS and 32 KW of fast static RAM memory. The array is connected to an "external host" that has a large (from 8 to 59 MB) memory and two MC68020 "cluster processors" that can feed data to and from the Warp array. (There is also a "support processor", which was not used in this benchmark). The external host is connected in turn through a bus repeater to a Sun 3/160 workstation. Programs on Warp are executed using an "attached processor" model; the user code on the Sun prepares data, then calls Warp for processing, which may take place while the Sun continues to do other operations.

An integrated implementation of the Warp machine called iWarp is underway as a joint project between Carnegie Mellon and Intel Corporation (Borkar, et al, 1988). The heart of an iWarp system is the iWarp component: a single chip processor that requires only the addition of memory chips to form a complete system building block, called the iWarp cell. Each iWarp component contains both a powerful computation engine (20 MFLOPS) and a high throughput (320 MB/s), low latency (100-150 ns) communication engine for interfacing with other iWarp cells. An iWarp system can include a large number of cells; the initial iWarp demonstration system consists of an 8×8 torus of iWarp cells, delivering more than 1.2 GFLOPS. It can be expanded to include up to 1,024 cells.

There are two programming languages that are used on Warp for computer vision. The first, W2, is the basic language for the array. W2 is a Pascal-like language in which each individual cell is programmed, and send and receive statements are used to transfer data between adjacent cells. W2 provides basic data and control structures such as integer and floating-point scalars and arrays, conditionals, and looping statements.

The second, Apply, is a specialized language for local operations in computer vision (Hamey, Webb, and Wu, 1987; Wallace, Webb, and Wu, 1988). In Apply, the programmer writes a function that will be applied around a single pixel in an input image; the Apply compiler generates a program that causes this function to be applied in parallel all across the image. This makes it easy to write such local operations as edge detection, smoothing, graylevel transformations, and so on. Global operations like histogram and Hough transform cannot be written in Apply. The Apply compiler has been mapped onto a wide variety of parallel computers, and other implementations are in progress. On Warp, Apply generates a W2 program.

Apply and W2 have been used to create a library of about 140 programs called WEB that covers most local and global low-level vision operations. The library includes histogram, connected components, FFT, edge detection, smoothing, and so on. These are available as compiled programs to be executed on Warp, or as models to follow to create new vision operations.

CONNECTED COMPONENT LABELLING

The connected components algorithm was implemented by H. Printz (Deutch, et al., 1988; Kung and Webb, 1985) of Carnegie Mellon. It uses a divide an conquer technique where slices of the image are first labelled independently on each cell, then the labels are unified by examining the borders between the cells. The merge is performed in a sequential pass; cell 0 merges its labels with cell 1, then cell 1 with cell 2, and so on. Finally, each cell passes its labelled image forward through the array where it is relabeled by the following cells, if necessary, to merge regions.

We did not attempt to use the recommended algorithm in this step, because we had existing code, and because the recommended algorithm would probably perform much worse than the Printz algorithm. The recommended algorithm propagates labels within the image without using an equivalence table. Each pixel is given a unique label based on its row and column coordinates, then labels are propagated from one pixel to another if both pixels have the same grayvalue in the original image. By performing repeated propagations in different directions, all pixels can be correctly labelled.

Depending on the order in which different directions are chosen, the worst case pattern can change, but for any choice of directions there is a pattern that requires this algorithm to make a number of steps on the order of the area of the image. (For example, if the different directions are always followed in the same order, a spiral pattern one pixel wide will take this long). This is very poor performance in the worst case. However, it was not clear from the algorithm description whether worst case behavior was an issue. In the test images, all of the interesting regions were compact and rectangular. If we ignore the background, only a few passes of propagation were necessary.

In any case, this algorithm is better suited to large processor arrays with simple processors than to small processor arrays with powerful processors like Warp. The only operations that are performed in the recommended algorithm are comparison of pixel values, and assignment of the smaller of two integer values. This means that computers that provide these facilities, and no others (such as address calculation or floating-point arithmetic) in their hardware will be at an advantage for this algorithm.

The structure of the algorithm also makes it well suited to large processor arrays. Each step involves transfer of a label value and pixel value to a pixel's neighbor. The time for this step is $T \times I/P$, where T is the time for the computation for a single transfer and computation, I is the number of pixels in the image, and P is the number processors. The best case is obtained when $I=P$, so that there are as many processors as image pixels – a very large number. For Printz's algorithm, the execution time is $AP+B/P$, where A is the time to do a merge step, and B is the time to label the entire image serially. The best case is obtained when $P=\sqrt{B/A}$. With the parameters taken from the first DARPA IU benchmark, this gives $P{=}16$, which is much closer to the Warp machine's actual number of processors.

The connected components code consisted of 226 lines of W2 code. (Because the problem to be solved involves a global computation, only W2 could be used to program it on Warp). Of the 226 lines, 60 were in the labelling of the image locally, and 62 were in the computation of the global maps. The remaining 104 statements were declarations and I/O statements to read the image into and out of the Warp array.

A comparable program for a serial computer written in FORTRAN, CLAB from the Spider library (SPIDER, 1983), consists of 59 lines. The W2 code is thus nearly the same length (in its serial section, i.e., labelling the image slices individually) as the FORTRAN code. This reflects the similar level of the W2 and FORTRAN languages. The merge operations needed because of parallelism double the total code, and additional I/O statements and declarations double it again.

TRACING AND COMPUTING K-CURVATURE

This step could be broken down into two steps: (1) computing the direction numbers of boundary pixels in the labelled component image; (2) extracting the boundaries, smoothing them, taking their derivative, and detecting zero crossings. The first step could be done purely locally, since the direction number of a pixel depends only on the pixels in a 3×3 neighborhood around it. We implemented step (1) in Apply, and used the provided C code to perform the rest of the computation.

The Apply code was a straightforward translation from the provided C code. We show the Apply program In Figure 2 to illustrate its simplicity. For comparison we show the original serial C code in Figure 3. Note that the Apply program is shorter. Because the Apply compiler automatically handles border processing in a reasonable way, it is not necessary to include the `lok`, `tok`, etc. tests in the user program. Also, Apply makes image pixel references relative to the current position, which eliminates the x and y variables in the C program.

```
procedure set_chain(intensity : in array(-1..1,-1..1) of real,
                    ch        : out byte)
is
   cval: integer;
begin
   ch := 0;
   cval := intensity(0,0);

   if intensity(-1, 0) = cval then ch := ch | 1; end if;
   if intensity(-1, 1) = cval then ch := ch | 16; end if;
   if intensity( 0, 1) = cval then ch := ch | 2; end if;
   if intensity( 1, 1) = cval then ch := ch | 32; end if;
   if intensity( 1, 0) = cval then ch := ch | 4; end if;
   if intensity( 1,-1) = cval then ch := ch | 64; end if;
   if intensity( 0,-1) = cval then ch := ch | 8; end if;
   if intensity(-1,-1) = cval then ch := ch | 128; end if;

end set_chain;
```

Figure 2: Apply Code for Computing Direction Numbers

The second step of the code was executed using the provided Sun C code. This included a scan of the direction number image in order to trace boundaries, followed by three convolutions of the boundary vectors to detect corners.

This step could have been made faster by a Warp implementation. There were two reasons we did not use Warp: (1) The boundary tracing operation is difficult to do in parallel. The image could be split up among processors, or the boundaries could be distributed to processors. In the first case, a separate step is required to merge boundaries, which can be quite difficult. In the second case, locating the boundaries in the image and making sure that different processors have different boundaries requires a second component labelling pass over the image; and memory allocation is difficult, since each Warp cell does not have enough memory to store the whole image. (2) The boundary tracing could be done in parallel on the Sun with other steps in the system, leading to a lower overall time for the system if it was not done on Warp.

MEDIAN FILTER

Depending on the problem, either 3×3 or 5×5 median filtering could be required. We had existing 3×3 median filter code already in WEB, which had be heavily optimized. The 5×5 code was written new for the benchmark.

3×3 MEDIAN FILTER

The 3×3 median filter maintains three sorted column lists, only one of which must be updated as the window is shifted to the left. The three column lists are sorted according to their middle element, forming what we will call the "smallest," "middle," and "largest" list. The smallest two elements of the smallest list are smaller than at least five elements of the 3×3 window, so they cannot be the median. Similarly for the largest two elements of the largest list. This leaves five elements; these are sorted. The median of these is the median of the 3×3 window.

In the Warp implementation of this algorithm the image is divided by columns into ten slices, each cell taking one slice, and the medians are calculated independently in each cell as the image is fed to the array row by row. This

```
static unsigned char set_chain(intensity,x,y)
bmark_byte_image intensity;
int x, y;
{
register unsigned char ch;
register int cval,lok,tok,rok,bok;

  ch = 0;
  cval = intensity[y][x];

  /* set boundary flags */

  lok = x > COL_MIN;
  tok = y > ROW_MIN;
  rok = x < COL_MAX;
  bok = y < ROW_MAX;

  /* 4-connected boundary  & 8-connected boundary */

  if (tok) {
    if         (intensity[y - 1][x    ] == cval)  ch = ch |   1;
    if (rok && (intensity[y - 1][x + 1] == cval)) ch = ch |  16;
    }
  if (rok) {
    if         (intensity[y    ][x + 1] == cval)  ch = ch |   2;
    if (bok && (intensity[y + 1][x + 1] == cval)) ch = ch |  32;
    }
  if (bok) {
    if         (intensity[y + 1][x    ] == cval)  ch = ch |   4;
    if (lok && (intensity[y + 1][x - 1] == cval)) ch = ch |  64;
    }
  if (lok) {
    if         (intensity[y    ][x - 1] == cval)  ch = ch |   8;
    if (tok && (intensity[y - 1][x - 1] == cval)) ch = ch | 128;
    }

  return(ch);
}
```

Figure 3: C Code for Computing Direction Numbers

partitioning method wastes computation since each cell has to reinitialize the median filter for each row, as opposed to partitioning by row, where only one cell has to reinitialize the median filter for each row. It has the advantage of allowing computation of results on line, and requiring less memory per cell (which was the overriding factor when this code was implemented on the prototype Warp machine, which had only 4K words of memory per cell.)

5×5 MEDIAN FILTER

While median filter can be formulated as a local operation, the provided C code used raster order processing to speed up execution time. This made it impossible to implement the code using Apply, which does not allow the programmer to take advantage of raster order processing. Therefore, the 5×5 median filter algorithm was a straight translation of the provided C code into W2. The image is divided into 10 horizontal slices, with each processor getting one slice. To compute the first median in a row the values in the 5×5 region are sorted and the 13th largest is chosen. To compute the next median in the row five new values (the right side of the 5×5 region) are inserted into the sorted list, and the five values no longer in the region are removed from the list. The 13th is then chosen as the median for this window. This process repeats for the rest of the elements in the row.

The programming effort for this algorithm was minimal, since the partitioning the image by rows is straightforward, and processing of a row is done exactly the same way as in the C code. This is illustrated by the statement counts for the different parts of the code; 28 W2 statements were involved in programming the image I/O, and 52 statements were needed for the computation, versus 67 statements for computation in the C code.

GRADIENT MAGNITUDE AND THRESHOLDING

The Sobel operator was a straightforward implementation of the provided C code. The Apply code for the Sobel operator is shown in Figure 4. The code development time for this program was only 10 minutes.

```
procedure grad(imagein  : in array (-1..1, -1..1) of real,
               threshsquared : const real,
               imageout : out byte)

is

   xderiv, yderiv : real;

begin

   xderiv := 0.25*imagein(-1,-1) + 0.5*imagein(-1, 0) + 0.25*imagein(-1, 1) -
           0.25*imagein( 1,-1) - 0.5*imagein( 1, 0) - 0.25*imagein( 1, 1);

   yderiv := 0.25*imagein(-1,-1) + 0.5*imagein( 0,-1) + 0.25*imagein( 1,-1) -
           0.25*imagein(-1, 1) - 0.5*imagein( 0, 1) - 0.25*imagein( 1, 1);

   if (xderiv * xderiv + yderiv * yderiv) < threshsquared
        then imageout := 0;
        else imageout := 255;
   end if;

end grad;
```

Figure 4: Apply Sobel Operator

DEPTH AND HOUGH PROBES

The depth and Hough probes were performed as part of a graph matching algorithm that ran on the Sun. The probe routines were compact loops that consumed the majority of the computation time for the graph matching. Moving these loops to Warp was straightforward, since they consisted of operations that accessed a rectangular area of the image, and performed a few simple tests on it.

DEPTH PROBE

The depth probe took place in a bounding box aligned with the horizontal and vertical image axes in which a rectangle was supposed to lie. Pixels within the region were first tested to see if they fell within the supposed rectangle (which could have any orientation) and then, if they did, one of several counters was incremented depending on whether the pixel was deeper or closer than the supposed rectangle, or at the right distance and with the right intensity value.

Since the processing of each pixel was independent, there was no advantage in dividing the region in some large-grain fashion, such as splitting it into adjacent strips of columns. Moreover, there was considerable overhead in doing this, because the padding needed to make the image divide evenly into strips would vary for each region, and would have to be computed at run time, since the bounding box size was determined then. Also, the loops distributing the image across the cells would have bounds that could be known only at run time, so that the compiler could not take advantage of known loop bounds to perform various optimizations, such as pipelining. Instead, the region was distributed across the ten cells by having each cell take every tenth pixel as the region was sent in raster order from the cluster processor. Once the entire region was sent, the cluster processor sent a series of sentinels that caused the cells to total and output their counters.

Code download was done only once per execution. It averaged 1.6 s with a standard deviation of 1.3 s. The standard deviation of the code download time is high because the system software must verify, before each probe, that the code has already been downloaded and must also initialize the machine state. This time is included in the code download time, but varies with the number of probes. A linear regression model of code download time of the form $a+bN$, where N is the number of probes, gives a=185 ms, and b=11.3 ms. In other words, the initial code download takes 185 ms, and the incremental cost (in overhead counted as code download time) for each probe is 11.3 ms.

HOUGH PROBE

While the image region to be processed could vary, the Hough space produced was fixed. Therefore, we chose to split the Hough space among cells, and have each cell process the entire image. This led to a straightforward W2 program, the inner loop of which is shown in Figure 5.

Each cell receives the pixel, and forwards it to the next cell. The cell then updates its portion of the Hough space, which is divided by theta values – each cell takes every tenth theta value. The calculation of `rho` is as in the C code. Once all pixels are processed, the cells concatenate the Hough space and output it to the host.

```
FOR i := strow TO endrow-1 DO BEGIN
  y := i - MID_ROW;
  FOR j := stcol TO endcol-1 DO BEGIN
    x := j - MID_COL;
    RECEIVE(l, x, ipixel); SEND(r, x, ipixel);
    IF ipixel <> 0 THEN BEGIN
      htp := 0;
      FOR theta_offset := 0 TO 80 BY 10 DO BEGIN
        theta := theta_offset + cellid;
        rho := fix((x*sin[theta] - y*cos[theta]) / RBINSIZE);
        IF rho < 0 THEN
          BEGIN posx := htp+9; posy := 0-rho; END
        ELSE
          BEGIN posx := htp; posy := rho; END;
        hough[posx][posy] := hough[posx][posy] + 1.0;
        htp := htp + 1;
      END;
    END;
  END;
END;
```

Figure 5: The inner loop of the W2 Hough program

The code was downloaded only once, with an average time of 900 ms, and a standard deviation of 748 ms. The standard deviation of Hough probe code download time is high for the same reason as with match strength probe. The linear regression model gives the initial code download time as 179 ms, and the extra time per probe is 13.9 ms. These times are consistent with those for match strength probe.

PAINT RESULT

After the model was detected, it was painted over the input image and the resulting image was output. In the provided C code the paint rectangle routine was called repeatedly as the model graph was traversed. This would have considerably slowed the Warp implementation, since the overhead for program startup is significant. In our implementation, the rectangle descriptions are stored in a list, which is then passed to the Warp for painting all at once.

Two different implementations of the paint rectangle routine were tried. In the first Apply code was used. All rectangle descriptions were passed as parameters to the Apply routine. The Apply program checked to see if its pixel fell within the rectangle description, and if it did then it was painted. (The rectangle descriptions were ordered in the same order as the C code would have painted them. Hence the Apply code always produced the same result as the original C code, even when rectangle descriptions overlapped.)

In the second implementation, which was written in W2, each cell took one tenth of the rectangle descriptions. A row of the image was passed in, and each cell painted its portions of the rectangles onto that row, passing the painted row on to the next cell. As before, the rectangles were ordered so that painting happened in the same order as the original C code.

The second implementation was faster, mainly because some of the testing could be done on a per-row basis (namely, whether the row intersected with the circumscribing rectangle aligned with the image axes around the rectangle to be drawn). In the Apply code this had to be done on a per-pixel basis.

The time for this routine was 2.3 s with a standard deviation of 179 ms. Of this, approximately 97% (standard deviation 0.40%) of the time was spent in the Warp paint rectangle routine.

EXPLOITING TASK-LEVEL PARALLELISM

Warp is programmed using the *attached processor* model, in which a general purpose host (a Sun workstation) runs the majority of the code, which is also the least time-consuming portion, and calls the Warp array to do time-consuming small portions of the program. In this benchmark, these portions of the code are those that access images. Almost every operation that touches all or a large portion of the pixels in the image runs on Warp. (The only exception is the boundary extraction code).

We take advantage of the intrinsic parallelism in this model at several points in the benchmark. Warp can run on its own, doing some image processing, while the Sun does other tasks in parallel.

Exploiting task-level parallelism in this way allows us to largely eliminate the overhead of reading in images during the benchmark, which would otherwise constitute a significant portion of the total benchmark time. Figure 6 shows where we were able to make use of task-level parallelism: that is, in "read depth image," "read model descriptions," and "extract strong cues." "Read depth image" involves reading the floating-point depth image from disk, and "read model descriptions" involves reading the model descriptions in, where they are stored in ASCII. Because the depth image is so large, it takes about one second to read; since the model descriptions are stored in ASCII, they also take about one second to read and translate into internal binary format.

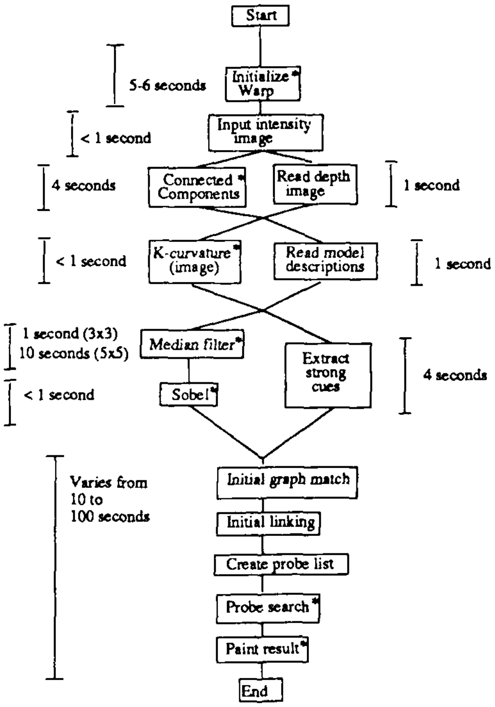


Figure 6: Task-level parallelism in the IU Benchmark

Because the depth image and model data are read in parallel with a Warp computation, they do not contribute to the total benchmark time; their time is subtracted from the benchmark time. This effect is better than the normal speedup provided by faster hardware, which is only multiplicative.

While the depth image is being subjected to median filter and the Sobel operator on Warp, "extract strong cues" runs in parallel on the Sun. ''Extract strong cues'' is a complex procedure that is difficult to parallelize, and which would have required a lot of reprogramming to run on Warp. Moreover, since it could be run in parallel, its execution did not contribute to the total benchmark time (at least in the case when 5×5 median filter was used).

Exploiting task-level parallelism in this way is a powerful and general method of reducing total program time, because it does not rely on particular hardware features; instead, any routine that does not have dependencies with later routines can be executed in parallel. Task level parallelism is therefore much easier to exploit than data parallelism, which was exploited in the routines implemented on Warp; only minimal reprogramming is necessary. Unfortunately, there are only a few places in the benchmark where task level parallelism can be used. This is due to the general nature of recognition programs, where almost every reasoning step depends on those preceding it.

PERFORMANCE SUMMARY

Table 1 gives all the performance figures for Warp on the IU benchmark. The times given for indented lines are included in the first non-indented line above. Means and standard deviations of execution times are also given. All times are in seconds. All numbers are rounded to three significant digits.

The startup time for Warp programs was estimated to be about 25 ms, not counting code download time, which is given in the table.

MAKING PARALLEL PROGRAMMING EASIER

In the implementation of the Warp routines above, there is a strong difference between those implemented using W2 and those implemented using Apply. The Apply routines are shorter than the corresponding C programs, and generally took only a few minutes to program; the W2 routines are longer than the C routines, and generally required days of careful programming to get right. Here we examine the reasons for this difference.

The W2 programmer has to do several tasks that the Apply programmer avoids:

1. Input and output the data structures from Warp.

2. Sequence the operation across the input data structure. The C programmer must also do this step.

3. Combine separate portions of the output data structure into one structure.

The W2 programmer is also at one other important disadvantage compared to the Apply programmer; W2 code runs only on Warp, while Apply programs run on several machines, with more implementations underway. It is far easier to exchange Apply programs between machines, making Apply programs more useful as a medium for the exchange of ideas, and also Apply programs have a much longer useful lifetime than W2 programs. (For example, the Apply programs in WEB have been ported from Warp to iWarp and through two changes in Warp/W2 architecture without change, while the W2 programs needed or will need changes in all three cases.)

The W2 programmer, however, has two critical advantages compared with the Apply programmer: (1) W2 is a general language, and (2) It is possible to exploit raster-order processing in W2, giving greater efficiency. The first advantage is the reason that connected components and the Hough probe algorithm were written in W2; the second is the reason W2 was used for median filter and the paint result routine.

We now consider whether it is possible to extend Apply so that it can be made more general, and capable of exploiting raster-order processing, without losing the advantages listed above.

Let us make a list of requirements for the new, extended Apply:

1. It should be capable of raster-order processing for greater efficiency.

2. It should be capable of computing global image processing algorithms.

Considering the global image processing algorithms in this benchmark, we observe that they all order their processing in the following way:

- The image is broken down into sections, one section per processor.

Step	Sample	Test2	Test3	Test4	Test	Mean	Standard deviation
Task startup/initialization (initializing Warp)	5.76	5.96	5.88	6.00	6.04	5.93	0.111
Image input: intensity image	0.780	1.22	1.24	1.22	0.700	1.03	0.268
Image input: depth image	2.74	4.18	4.10	4.12	3.02	3.63	0.694
Model input	1.30	1.02	1.08	1.06	1.18	1.13	0.113
Label connected components	3.98	4.60	4.54	4.56	4.04	4.34	0.306
Warp code download time	0.400	0.420	0.440	0.420	0.420	0.420	0.0141
K-curvature	3.14	2.20	2.72	2.54	2.24	2.57	0.385
warp code download time	0.260	0.260	0.260	0.240	0.260	0.256	0.00894
Smooth K curvature	1.38	0.780	0.980	0.900	0.640	0.936	0.279
First derivative of curvature	0.420	0.280	0.340	0.400	0.240	0.336	0.0767
Zero-crossing detection	0.320	0.120	0.140	0.220	0.060	0.172	0.110
Final corner detection (includes threshold and AND)	0.160	0.120	0.220	0.200	0.100	0.160	0.0501
Count corners for each component and threshold	0.0200	0.0400	0.0600	0.0600	0.0200	0.0400	0.0200
Convex hull of components	0.0200	0.0200	0.0800	0.0600	0.000	0.0360	0.0329
Test for sequence of three right angles	0.000	0.0200	0.0200	0.0200	0.000	0.0120	0.0110
Final rectangle hypothesis generation	0.0400	0.0200	0.0200	0.0400	0.000	0.0240	0.0167
3×3 Median filter		1.38	1.40	2.00		1.59	0.352
5×5 Median filter	10.7				8.70	9.70	1.41
Warp code download time	0.280	0.300	0.280	0.880	0.280	0.404	0.266
Sobel (includes conv., grad. mag., threshold)	0.480	0.720	0.940	0.920	0.480	0.708	0.225
Warp code download time	0.320	0.320	0.340	0.320	0.320	0.324	0.00894
Initial graph match	0.420	0.220	1.22	1.38	0.240	0.696	0.560
Rectangle matching	0.200	0.160	0.400	0.680	0.160	0.320	0.224
Initial linking	0.220	0.060	0.820	0.700	0.0800	0.376	0.358
Match extension	24.8	4.58	38.6	41.2	3.64	22.6	18.0
Match-strength probe	9.10	2.86	13.6	13.5	2.64	8.34	5.42
Total probes	91	20	247	239	23		
Window selection	0.0200	0.0200	0.240	0.180	0.0200	0.0960	0.106
Classification and counting	9.00	2.82	13.2	13.1	2.56	8.14	5.25
Time per probe	0.0989	0.141	0.0534	0.0548	0.113	0.0922	0.0380
Warp code download time	1.18	0.460	2.84	3.08	0.420	1.60	1.28
Hough probe	15.3	1.68	23.3	25.8	0.960	13.4	11.7
Total probes	58	5	97	95	3		
Window selection	0.0200	0.0200	0.120	0.0600	0.000	0.0440	0.0477
Hough transform	12.8	1.44	19.3	20.0	0.880	10.9	9.31
Warp code download time	0.840	0.260	1.06	2.08	0.260	0.900	0.748
Edge-peak detection	2.38	0.220	3.80	5.58	0.0800	2.41	2.36
Rectangle parameter update	0.0200	0.000	0.000	0.0800	0.000	0.0200	0.0346
Result presentation	2.60	2.52	2.24	2.26	2.26	2.38	0.171
Best match selection	0.0200	0.000	0.0200	0.0200	0.000	0.0120	0.0110
Graph traversal and image generation	2.54	2.46	2.16	2.18	2.20	2.31	0.178
Warp code download time	0.280	0.280	0.300	0.280	0.280	0.284	0.00894
Total execution time	52.9	29.9	65.8	69.3	29.4	49.5	19.1

Table 1: Performance of Warp on the IU Benchmark

- Each processor computes a global result on its section.
- The global results are combined to create the global data structure for the whole image.

Our experience suggests that this *divide and conquer* approach to global image operations is useful and general. Therefore, we will use it in the extended Apply as the uniform paradigm for global image operations. We will add to Apply the ability for the programmer to define a special *combining* function; this takes the output of two Apply functions applied over areas of the image and combines them. For example, the combining function for histogram is the operation of adding the two histograms. We also add a *termination* function, which is applied once to the final global data structure; this can discard intermediate results needed for the combining operations and produce the global output. In order to initialize the data structures that are being computed, we add an *initialization* function.

We will also add raster-order image processing to Apply. An important issue is whether this would make it more difficult to implement Apply (since restricting the programmer to order-independent operations was supposed to make Apply compilers easier to implement). Our experience suggest that it does not; all Apply compilers (except for the one for SLAP (Fisher and Highnam, 1987)) process the image in raster order anyway. (Raster order processing makes sense only on processors that have much fewer processors than pixels. This is the reason it would be hard to implement efficiently on an architecture like SLAP. The same problem exists on other architectures, for example the Connection Machine (Tucker and Robertson, 1988).)

Raster order processing implies that there must be some way of initializing the computation, since raster order algorithms assume some previous state. The initialization function we added in order to compute global operations can serve this purpose. The current Apply function becomes the function that is applied repeatedly, in raster-order, across the image. Note that the presence of an initialization function allows us to start the raster-order processing anywhere in the image; this makes it possible to still process the image in parallel, with raster-order processing at each cell.

Raster order processing allows us to think in terms of processing regions of pixels. Because of this, we can add a useful restriction to the applications of the combining function; we require that Apply use it to combine global variables computed only over adjacent regions of the image. This makes it easier to implement many global image operations, for example connected components.

To summarize, extended Apply programs consist of up to four parts:

1. An initialization function, which can be run anywhere in the image.
2. A raster-order function, which is applied in raster-order across the image (wrapped around the borders of the image). The first execution of the raster-order function is guaranteed to be preceded by an execution of the initialization function.
3. A combining function, which combines the outputs of any two image regions to produce an output for the concatenation of the two regions. In order to make programming easier, we stipulate that the combining function will be applied only to adjacent "swaths" (groups of consecutive rows) of the image.
4. A termination function, which is applied once after the output of the entire image is computed.

Of course, it is not necessary to have all these parts in an Apply program; only the raster-order function is required.

To illustrate how the Apply compiler could use these functions on different parallel computers, let us call the initialization function I, the raster-order function R, the combining function C, and the termination function T. On a serial processor, these functions will be executed as follows on a $N \times N$ image:

$$I(), R(0,0), R(0,1), \ldots, R(0,N), R(1,0), \ldots, R(N,N), T()$$

Subscripts are used to represent the image pixel at the center of the window processed by the function.

On a multiprocessor where the image is broken down horizontally, each processor taking an adjacent set of rows, a similar program will be used, except that each processor will process only its rows, and the combining function will be applied to merge results from adjacent stripes. One processor's program might look like this:

$$I(i,0), R(i,0), R(i,1), \ldots, R(i,N), R(i+1,0), \ldots, R(j,N), C()$$

where the application of $C()$ merges this processor's results with the results from some other processor. The order of the combining operations depends on the interprocessor communications facilities.

If we partition the image vertically, each processor taking an adjacent set of columns (as in the Apply Warp implementation) we can still implement raster order processing by passing intermediate results between processors after each has completed processing its portion of the columns. The program on the first processor looks like this:

$I(i,0), R(i,0), \ldots, R(i,j), Send()$

where $Send()$ sends the intermediate results from this processor to the next. Intermediate processors have this program:

$Receive(), R(i,k), \ldots, R(i,l), Send()$

and the last processor has this program:

$Receive(), R(i,k), \ldots, R(i,l), C()$

Processing the image in this way creates a pipeline of processors, staggered diagonally across the image. After processor 1 has started processing the first row, processor 0 goes on the process the second. This mapping would be efficient only on computers with low communications overhead, such as Warp.

There are several ways of merging regions computed on different processors. With only nearest-neighbor communications in a linear processor array, regions would be merged serially – i.e., the first region is merged with the second, this result is then merged with the third region, and so on, resulting in a tree of merge operations like that shown in Figure 7. On a machine that has more flexible interprocessor communications facilities, regions can be merged in parallel as shown in Figure 8.

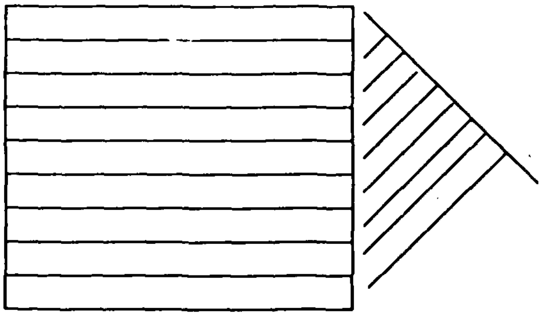


Figure 7: Merging results from adjacent image regions serially

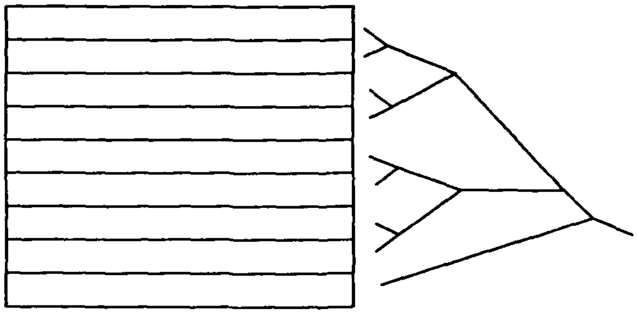


Figure 8: Merging results from adjacent image regions in parallel

The image can be broken down differently, for example by columns, or by allocating a square region to each processor. Dividing the image in such a way as to give each processor a complete row has the advantage that *Send*() and *Receive*() functions are unnecessary. However, breaking the image down in other ways also has advantages. Dividing the image by columns allows us to compute local Apply operations on-line; as each row is fed to the processor array, another row of results is computed. Dividing the image into squares allows us to use more processors efficiently, since Apply programs must duplicate processing at the region perimeter, and the rectangular region of a given area with the shortest perimeter is a square.

There is an important limitation in the divide and conquer when we try to reduce execution time by using more and more processors. Because of the overhead of the merge step, execution time decreases until a certain point, after which using more processors actually results in longer execution time. The number of processors that gives the shortest execution time depends on the algorithm, the image size, and the interprocessor communication method. Let A be the time to merge two adjacent regions, and B be the time to process the entire image in raster order. With serial merging of regions as in Figure 7, the optimal number of processors is $\sqrt{B/A}$; on a two-dimensional array of cells with two merge steps (one horizontal and one vertical) the optimal number is $(B/A)^{2/3}$; and with the merge steps implemented as in Figure 8, each merge halving the number of regions to merge, the optimal number is $B\ln 2/A$. Table 2 gives the optimal number of processors for three algorithms: image sum, histogram, and connected components, all on 512×512 and 10,000×10,000 images. The time for 512×512 connected components comes from its Warp implementation in the first DARPA Image Understanding benchmark; the other times are estimated.

We also give the "knee" of the execution time profile, where the benefit (increase in speedup) per unit cost (decrease in efficiency) is maximized (Eager, et al., 1989). Defining the efficiency of the parallel implementation on N processors as $E_N=T_1/(NT_N)$, where T_k is the execution time on k processors, this point is the k for which E_k/T_k is maximized. The point at which this happens is the most cost-effective number of processors to use. For a linear array, this number of processors at the knee is $\sqrt{B/3A}$; on a two-dimensional array, it is $(B/4A)^{2/3}$. There is no closed form solution for the number of processors with parallel merging; it is N in the equation $N=(B/A)\ln 2/(\ln N+2)$.

Algorithm	Image size	B/A	Serial merge		Two merge steps		Binary tree merge	
			Max	Best	Max	Best	Max	Best
Image sum	512×512	262,000	512	296	4100	1625	181,000	15,600
	10*K*×10*K*	100,000,000	10,000	5773	215,000	85,499	69,300,000	4,030,000
Histogram	512×512	1020	32	18	102	40	710	106
	10*K*×10*K*	391,000	625	361	5340	2120	271,000	22,500
Connected components	512×512	256	16	9	40	16	177	32
	10*K*×10*K*	5000	71	41	292	116	3470	430

Table 2: Global operations with different communications methods

Table 2 gives us some justification for allowing the programmer to assume that the image is divided only by rows when writing the combining function. For algorithms as complex as histogram, the overhead of the merge operation is great enough so that there is not enough parallelism available to require dividing the image in this way. The only cases where the available parallelism exceeds the number of rows are image sum with two merge steps or parallel merging, and histogram for binary tree merge (and even there the most cost-effective number of processors is near to or less than the image height). Even if we allowed image subdivision by columns, the Apply programmer and compiler could not make good use of it except with extremely simple merge operations. (Of course, in the absence of a combining function the Apply compiler can still divide the image by columns, without introducing any extra overhead, just as the current Apply does in its implementation on Warp.)

BENCHMARK GLOBAL IMAGE PROCESSING OPERATIONS IN APPLY

We now consider how the global image processing operations in this benchmark can be written in the extended Apply.

We will sketch the implementations of each algorithm and avoid detailed questions of their implementation in the new language. We order the algorithms in order of difficulty.

MEDIAN FILTER

The median filter algorithm can be written using a raster-order function as follows:

- At the beginning of each row, the raster-order function sorts the elements in the window and stores them in a list. It then selects the median element from this list; this is the output. In preparation for moving the window to the right, the leftmost elements of the window are deleted from the list, and the new rightmost elements are added.

Since the median filter algorithm does not compute a global result, there is no need for combination or termination functions.

PAINT RECTANGLE

As supplied with the benchmark, the paint rectangle algorithm performs an important optimization; for a given rectangle, it determines from its bounding box whether or not the current row of the image intersects with the rectangle or not. Our extended Apply program can make the same optimization:

- At the beginning of each row the raster-order function determines, for the current row, whether or not it intersects each of the rectangles being painted. It then goes through each of the intersected rectangles in order (from most distant to nearest), determining if the current pixel lies within them, and repeatedly assigning the current pixel the color of the rectangle within which it falls. The result is that the current pixel gets the color of the nearest rectangle containing it.

DEPTH PROBE

The depth probe operation as supplied with the benchmark processes only with the bounding box of the rectangle that is being probed. It computes a simple global operation on this rectangle:

- The initialization function zeroes the various counters: points too deep, points at the correct depth, and other points. It also zeroes a variable that keeps track of the sum of depths of points at the correct depth.
- The raster order function first determines if a pixel falls within the rectangle. If it does, it updates the appropriate counter, and adds its depth to the sum of depth variable if it is at the correct depth.
- The combination function simply adds together corresponding variables from the adjacent regions.
- The termination function calculates the average depth by dividing the sum of depth variable by the total number of correct points.

Note that the W2 program implemented for the benchmark used a different method of partitioning the image than any we have proposed for implementing extended Apply programs. The image was dealt out to the processors, one pixel at a time, with no concern for how the processors mapped onto the image; each of the ten processors simply took every tenth pixel. This was done because there is no dependence whatsoever between adjacent pixels in the code as supplied with the benchmark, and because it was easier to program.

However, the extended Apply makes it possible to propose a refinement in the algorithm, which can also be used in the paint rectangle routine. For each row, it is possible to calculate the starting and ending pixels in that row that fall within the rectangle; this computation can be done in the raster-order function. The resulting code avoids having to test every pixel for whether it lies in the rectangle. Since this test is difficult, involving several floating point operations, the resulting code is faster, and no more difficult to implement in the extended Apply.

HOUGH PROBE

Hough probe performs a Hough transform on the Sobel-processed depth image to find rectangle edges. The region of the image processed is localized, but the Hough transform produced is not, although only a small portion of it is actually used. Using the divide and conquer model in this way has serious implications for memory usage. We present two different implementation of Hough probe in the extended Apply. In the first, we divide the image:

- The initialization function zeroes the Hough space and also calculates the sine and cosine tables that are used to update it.

- The raster order function first determines if the pixel is non-zero. If it is, it increments all elements of the Hough space that are mapped by this pixel; this is a sine wave in the Hough array.
- The combination function sums the two Hough spaces from the adjacent regions to give the new Hough space.

There is no termination function because the final output of this algorithm is the Hough space. However, we could implement more processing in the Hough probe by including some of the probing for weak and strong edges in the termination function. This would make it unnecessary to output the Hough space.

This implementation of Hough probe requires each processor to store the complete Hough space; worse, when the two Hough spaces are combined one processor must store two of them. This is unfortunate, since in parallel processors memory is usually at a premium. The Hough space used here is 180×512; this would not fit, for example, in the Warp cell's 32KW memory, or in iWarp's 128KW memory.

The W2 code implemented for the benchmark divides the Hough space among processors. We can do this in the extended Apply, by treating the output Hough space as an image. In this way, updating the Hough space becomes similar to a graphics operation, with the output Hough space being the output image:

- The initialization function calculates sine and cosine tables as before. (Since the initialization function is run at the beginning of a row, and the Hough array is indexed as (ρ,θ), we must initialize the complete sine and cosine table for each row).
- The raster order function need only consider, for a given Hough pixel, what image pixels map onto it. This is a line in the image; the raster order function indexes along this line, and increments the Hough pixel for each non-zero image pixel.

There is no combination or termination function, because we have formulated Hough transform as a *local* operation by treating the input image as a *global* parameter. This algorithm could actually be implemented in the current Apply as it is formulated here. (It was not implemented in this way for the benchmark because of technical restriction in the current Apply; global parameters must have dimensions that are know at compile time. The subimage processed by the Hough probe is determined at run time).

The second implementation of Hough probe has an advantage when only a small part of the Hough space is actually needed, as here (where strong and weak edges are searched for in part of the Hough space). Only that part of the image that maps on to the corresponding portion of the Hough space will be examined, if we precede the raster order function by a test if the Hough pixel is needed. In the first implementation of Hough transform, there is no good way of avoiding this unnecessary computation.

In this implementation, the input image is stored at all processors, while the Hough space is distributed. If the input image is large, it may not fit. In this case, it is possible to process the input image in slices, incrementing the Hough space appropriately for each slice. This results in some unnecessary computation (because the line parameters for the image scan have to be recomputed for each slice), but it can make the memory space used for each pass arbitrarily small.

CONNECTED COMPONENTS

One of the assumptions underlying the divide and conquer model implemented by the extended Apply is the idea of data reduction. The output data structure of this level of vision is assumed to be smaller than the input, which is an image. This assumption, which is true of most vision algorithms at this level, is not true of connected components, since the output data structure is an image, just like the input. This assumption is also violated by Hough transform, but as we have seen it is not difficult to deal with this problem in Hough transform, because of the many sources of parallelism there. It is harder to deal with it in connected components, however.

Connected components can be formulated in many different ways. These different methods have significantly different implications for the type of architecture that implements the algorithm most efficiently. The recommended algorithm for the benchmark is different from the algorithm we implemented on Warp; this is because the recommended algorithm is more suitable for a large processor array, while the algorithm we implemented is better suited for a small processor array like Warp. In this section we will examine both the algorithm we actually implemented and the recommended algorithm from the point of view of the extended Apply.

First, let us consider the algorithm we actually implemented. It breaks down into three parts:

1. Split the image into separate parts, allocating one region to each processor, and label them separately, building an equivalence table for each part.
2. By examining the boundaries between the parts, merge the equivalence tables.
3. Apply the merged equivalence table to the image, producing the completely labelled image.

The final step is actually done by application of the first cell's equivalence table to the entire image, followed by application of the second cell's equivalence table, and so on. Doing the applications in this way allows us to make only a single forward merge pass (from the first cell to the last) across the equivalence tables, instead of having to make both a forward and backwards pass, and also avoids having to create a unified global equivalence table.

We will put the first two steps in the above algorithm into one extended Apply program, and put the second step into a second program, since each requires a pass over the image. The first two steps in connected components can be written as follows:

- The initialization function zeroes the equivalence table.
- If the new pixel is non-zero, the raster order function gives it a label, copying the label from the pixel to the left, left and above, above, or right and above, if any of them are non-zero, and otherwise assigning it a new label. If two of these pixels are non-zero and have different labels, it performs a union between the two labels and assigns the smaller of them to the new pixel. (A standard UNION-FIND algorithm (Aho, et al., 1975) can be used, or see below.) The labelled pixels from the top and bottom boundaries of the image region are saved in a special buffer.
- The combining function merges the two equivalence tables by stepping along the touching boundaries, noting touching non-zero pixels, and performing a merge between the two equivalence tables.

There is an important optimization that can be performed in the raster-order function; because the image is two-dimensional, it is not necessary to perform a complete merge when two labels are merged. A region that is inside a second region can never be merged with any regions outside the second region. This optimization has been taken advantage of by several authors (Kung and Webb, 1985; Schwartz, et al., 1985).

The second step is a simple Apply program:

- The raster-order function sin.ply looks up the pixel in the equivalence table and outputs its mapping.

This implementation of connected components suffers from one flaw, which was partially overcome in the W2 program implemented for the benchmark; the number of region labels can be potentially very large, as much as one-quarter (for 8-connected components) to one-half of the image pixels (for 4-connected components). This makes the equivalence table very large, in fact much too large to fit in one Warp cell's memory. In practice, however, (and in this benchmark) the number of components is not this great.

The W2 program addressed this issue by maintaining separate equivalence tables on each cell, which were never completely merged. As long as an individual cell did not need to label more components than would fit in its table, the algorithm would work. We cannot use this approach in the extended Apply, since it maintains an idea of a single value of all global variables for the entire image.

This restriction shows one of the assumptions behind the divide and conquer model that we are using in Apply; that is that there is a reduction in the size of data being processed. Otherwise, the divide and conquer model becomes inefficient. In connected components this assumption is normally satisfied; but in extreme cases, it may not be, and connected components will have to be implemented directly in the target computer language, or in a different way.

Another way connected components can be implemented is as described in the algorithm supplied with the benchmark. In this algorithm, labels are simply propagated pixel to pixel, and no global equivalence table is used. The simplest way of implementing this is as an ordinary Apply program. Initially, we assign each non-zero pixel a label based on its position in the image, then

- The raster order function simply copies the minimum of the adjacent non-zero pixels and the current pixel to the current pixel.

This implementation is extremely inefficient, however. Each pass moves a label only one pixel, and the longest connected component in an image can have a length on the order of the area of the image; so the running time of the algorithm is proportional to the image area. By taking advantage of raster-order processing, we can cause this propagate labels all the way across one row or down one column in on pass, but this is not especially useful since propagating them in the other direction is just as hard as before.

THEORETICAL RESTRICTIONS OF THE EXTENDED APPLY

So far, we have taken a practical approach to understanding the class of image processing operations that can be computed in the extended Apply; we showed how several of the benchmark algorithms can be implemented in it. Now we consider the class of operations that can be implemented in extended Apply from a theoretical point of view.

Suppose we have a global operation $G()$ to be performed over an image $I=(a_0,\ldots,a_n)$. Let it be implemented by a series of applications of some raster-order operation $R()$:

$$G(I)=R(a_0,R(a_1,\ldots,R(a_n,\varnothing)))$$

$R()$ is the raster-order algorithm that is used to implement $G()$ on a serial computer. For example, if $G()$ is "calculate the histogram h()" then $R(a,h)$ is the operation "add 1 to h(a)," and the output of $R()$ is the new histogram.

On a parallel machine, we will execute a series of operations of $R()$ on different subsets of the image, then combine them somehow. We do this using a new function $C()$ such that

$$C(R(a_0,R(a_1,\ldots,R(a_{i-1},\varnothing))),R(a_i,R(a_{i+1},\ldots,R(a_n,\varnothing))))=R(a_0,R(a_1,\ldots,R(a_n,\varnothing)))$$

for any $0<i\leq n$. $R()$ is the raster-order function in extended Apply, and $C()$ is the combining function (we are ignoring the initialization and termination functions, which do not affect the proof.)

It turns out that for any local function $R()$, is is possible to define a function $C()$ that combines two outputs of $R()$ to produce the same result as repeated applications of a single instance of $R()$, so long as applying $R()$ to a sequence of data produces the same result as applying it in the reverse order.

The proof is as follows. Suppose we have the results of applying $R()$ repeatedly to two sequences A and B, and we want to compute the result of applying $R()$ to $A||B$. ($A||B$ is the concatenation of the sequences A and B). We write R^*A for the output of applying $R()$ to A, so we want to compute $R^*A||B$ given R^*A and R^*B. Given R^*B, we find (by enumeration, if necessary) a sequence C such that $R^*C=R^*B$. We then define

$$C(R^*A,R^*B)=R(c_0,R(c_1,\ldots,R(c_m,R^*A)))$$

Now we show that $C(R^*A,R^*B)=R^*A||B$. Consider the graph of applications of $R()$ that led to $R^*B=R^*C$, as shown in Figure 9. Below the point at which R^*B is computed, one branch of the graph leads down the application of $R()$ to the $\{c_i\}$, and another branch leads down the application of $R()$ to the $\{b_j\}$. We extend this graph upwards to the computation of $R^*A||B$ by applying $R()$ to the $\{a_k\}$. Now, since $R()$ is is reversible, we can invert the graph and get the same result $R^*A||B$ below c_0. But this result is exactly what we have defined as $C(R^*A,R^*B)$. This completes the proof.

This proof does not show that $C()$ can be computed efficiently. In extended Apply, the programmer is responsible for defining $C()$ reasonably. Note that if the programmer's definition of $C()$ computes the same result regardless of the order of application of $R()$, then $R()$ does not have to be reversible. Also, $C()$ may not compute the same result, but different results may be equivalent for the programmer's purpose. For example, floating point computations are not order independent, but generally the errors are small enough so that this does not matter. Or different orders of evaluation may calculate different results, but these might all be the same for the programmer's purpose–for example, the construction of the equivalence table in the connected components algorithm.

As we have seen, most global image computations can be formulated in this manner. Some cannot; these are image processing operations that depend on processing the image in a particular order, such as some half-toning

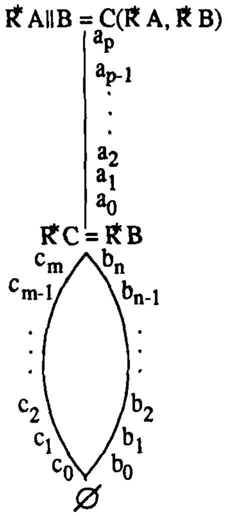


Figure 9: Graph of applications of $R()$ to compute $R^{*}A||B$

algorithms, the forward and backward pass in the two-pass grassfire algorithm, and certain region merging operations.

SUMMARY

In our implementation of the second DARPA IU benchmark, we explored a number of programming issues: choosing which routines to run on Warp, and which to run in parallel (when possible) on the Sun; choosing which to write in Apply, and which had to be written in W2; and choosing how to partition the data in the W2 programs.

We now believe it is possible to confront the issue of programming mid-level vision routines in largely machine-independent manner. We propose to use the divide-and-conquer programming model to do this.

We showed how the DARPA IU benchmark routines can be implemented in this model, and how flexible the model is for experimenting with different possible mappings of the routines. The theoretical basis of the model was explored, and it was shown that a general class of global image processing operations – those that are reversible – could be implemented with it.

ACKNOWLEDGMENTS

Many of the ideas for extending Apply to include global operations came from discussions with Leonard Hamey, and were developed in discussions with Claire Bono, C. H. Chien, and I-Chen Wu.

This research was supported in part by the Defense Advanced Research Projects Agency (DOD), monitored by the U.S. Army Engineer Topographic Laboratories under Contract DACA76-85-C-0002.

REFERENCES

Aho, et al., 1975. Aho, A., Hopcroft, J.E. and Ullman, J.D.. *The Design and Analysis of Computer Algorithms.* Addison-Wesley, Reading, Massachusetts, 1975.

Annaratone, et al., 1987. Annaratone, M., Arnould, E., Gross, T., Kung, H. T., Lam, M., Menzilcioglu, O. and Webb, J. A. "The Warp Computer: Architecture, Implementation and Performance". *IEEE Transactions on Computers C-36,* 12 (December 1987).

Borkar, et al, 1988. Borkar, S., Cohn, R., Cox, G., Gleason, S., Gross, T., Kung, H. T., Lam, M., Moore, B., Peterson, C., Pieper, J., Rankin, L., Tseng, P. S., Sutton, J., Urbanski, J. and Webb, J. iWarp: An Integrated Solution to High-Speed Parallel Computing. Proceedings of Supercomputing '88, IEEE Computer Society and ACM SIGARCH, Orlando, Florida, November, 1988, pp. 330-339.

Deutch, et al., 1988. Jeff Deutch, P. C. Maulik, Ravi Mosur, Harry Printz, Hudson Ribas, John Senko, P. S. Tseng, Jon A. Webb, and I-Chen Wu. Performance of Warp on the DARPA Architecture Benchmarks. International Conference on Parallel Processing for Computer Vision and Display, Leeds, England, January, 1988.

Eager, et al., 1989. Eager, D. L., Zahorjan, J., and Lazowska, E. D. "Speedup versus efficiency in parallel systems". *IEEE Transactions on Computers 38,* 3 (1989), 408-423.

Fisher and Highnam, 1987. Fisher, A. J. and P. T. Highnam. Communications, scheduling, and optimization in SIMD image processing. Computer Architecures for Pattern Analysis and Machine Intelligence, IEEE, 1987. (Submitted).

Hamey, Webb, and Wu, 1987. Hamey, L. G. C., Webb, J. A., and Wu, I-C. Low-level Vision on Warp and the Apply Programming Model. In *Parallel Computation and Computers for Artificial Intelligence,* Kluwer Academic Publishers, 1987. Edited by Janusz Kowalik.

Kung and Webb, 1985. Kung, H.T. and Webb, J.A. Global Operations on the CMU Warp Machine. Proceedings of 1985 AIAA Computers in Aerospace V Conference, American Institute of Aeronautics and Astronautics, October, 1985, pp. 209-218.

Rosenfeld, 1987. Rosenfeld, A. A Report on the DARPA Image Understanding Architectures Workshop. Image Understanding Workshop, DARPA, Los Angeles, California, February, 1987, pp. 298-301.

Schwartz, et al., 1985. Schwartz, J., Sharir M. and Siegel, A. An Efficient Algorithm for Finding Connected Components in a Binary Image. Robotics Research Technical Report 38, New York University, Courant Institiue of Mathematical Sciences, February, 1985.

SPIDER, 1983. Electrotechnical Laboratory. *SPIDER (Subroutine Package for Image Data Enhancement and Recognition).* Joint System Development Corp., Tokyo, Japan, 1983.

Tucker and Robertson, 1988. Tucker, L. W., and Robertson, G. G. "Architecture and Applications of The Connection Machine". *Computer Magazine 21,* 8 (August 1988), 26-38.

Wallace, Webb, and Wu, 1988. Wallace, R., Webb, J., and Wu, I-C. Architecture Independent Image Processing: Performance of Apply on Diverse Architectures. Third International Conference on Supercomputing, Boston, MA, May, 1988. To appear.

Weems, et al., 1988. Weems, C., Riseman, E., Hanson, A., and Rosenfeld, A. A Computer Vision Benchmark for Parallel Processing Systems. Third International Conference on Supercomputing, International Supercomputing Institute, Inc., Boston, MA, May, 1988, pp. 79-94.

Mean field theory for surface reconstruction

D. Geiger and F. Girosi
Artificial Intelligence Laboratory, MIT, Cambridge, MA 02139

Abstract

In recent years many researchers have investigated the use of Markov Random Fields (MRF) for computer vision. MRF have been generally used to obtain a reliable estimate of a function starting from a set of noisy data and some a priori knowledge about the function. These models can easily deal with discontinuities, introducing a discontinuity field (the so called line process) and the solution is usually found by means of Monte Carlo algorithms. The major drawbacks are the computational complexity of the implementation and the difficulty in estimating the parameters of the model.

In this paper we introduce a deterministic approach to two MRF models, based on some classical statistical mechanics tools. By studying the partition function of the system a set of non linear equations are obtained whose solution gives the reconstructed function and its discontinuities. We introduce the concept of *effective potential*, that allows us to eliminate the line process variable from the probability distribution. A study of the effective potential leads to a clear interpretation of the parameters of the model and then to a criterium for their estimates. The deterministic equation that we obtain are feasible of a fast, iterative and parallel solution.

We consider a model useful to reconstruct piecewise smooth functions, so that the set of data is smoothed almost everywhere but not at the discontinuities. The model is then improved to obtain a smooth discontinuity field and to enhance the contrast where a discontinuity appears. The improved model exhibits the concepts of threshold, suprathreshold and hysteresis for the detection of discontinuities. Parameter estimation is discussed and experiments with synthetic and real images are presented for both the models, confirming a better performance of the improved model.

1 Introduction

In recent years many researchers [8][11][5] [3][4] have investigated the use of Markov Random Fields (MRF) for early vision. MRF models can generally be used for the reconstruction of a function starting from a set of noisy sparse data, such as intensity, stereo, or motion data. Usually two fields are required in the MRF formulation of a problem: one represents the function that has to be reconstructed, and the other is associated to its discontinuities. The essence of the MRF model is that they give the probability distribution of the configuration of the fields, given a set of data, as a Gibbs distribution. The model is then specified by an "energy function", that can be modeled to embody the a priori information about the system. In the standard approach an estimate of the field and its discontinuities is given by the configuration that maximizes the probability distribution, or equivalently that minimizes the energy function. Since the discontinuity field is a discrete valued field (it assumes only the values 0 or 1) this becomes a combinatorial optimization problem, that can be solved by means of methods like the Monte Carlo one (simulated annealing[13], for example).

The MRF formulation is appealing because it allows to capture many features of the system of interest by simply adding appropriate terms in the energy function. However it has two main drawbacks: the amount of computer time needed for the implementation and the difficulty in estimating the parameters that control the relative weight of the various terms of the energy function.

In this paper we propose a deterministic approach to MRF models. It consists in explicitly writing down a set of equations from which we can compute, possibly by means of techniques of numerical analysis, estimates of the values of the field f and the line process. A natural framework of this approach is the equilibrium statistical mechanics, dealing with systems with many degrees of freedom described by Gibbs probability distributions. In principle statistical mechanics allows us to derive the mean statistical values of the field as *explicit* functions of the data and tha parameters of the model: an algorithm implementing this would

consist just of one step. Since the analytical computations required to obtain these explicit expressions are usually too hard, some approximations have to be made: as a result the solution is given in implicit form by a set of non linear equations, that we call *deterministic equations* to underline the deterministic character of the whole procedure.
We concentrate our attention on the partition function Z, that is the sum of the probability distribution over all the possible field configurations, since it is well known to contain all the information about the system. The idea underlving our approach is to first eliminate from Z the line process degrees of freedom: we will show that in doing so the effect of their interaction with the field can be simulated by a temperature dependent "effective potential" that depends only on f. Its use is fundamental in the derivation of the deterministic equations, and gives useful insights on the role and the significance of the parameters.
An advantage of such an approach is that the solution of the deterministic equations is faster than the Monte Carlo techniques, fully parallelizable and hopefully feasible of analog networks implementation. The possibility of writing a set of equations is useful for a better understanding of the nature of the solution and of the parameters of the model.

We discuss two different MRF models. The energy function of the first model has been already studied by several authors[1][10][12][11]. This energy function is not complex enough to introduce consistently concepts like smoothing with enhancement of discontinuities, hysteresis, threshold and suprathreshold.

In the second model we define an energy function with an extra term with respect to the previous one, establishing an interaction between the line process at neighborhood sites. This interaction can stimulate the creation of a line at a particular site if a line at a neighborhood site has been created. This term will allow the data to be smoothed, but, and at the same time, to enhance the contrast at the discontinuities. Typical edge detection features like hysteresis, threshold and suprathreshold, which do not arise from the previous energy function, will arise naturally from the model. We point out that this model is a generalization of the previous one, that can be recovered by simply setting an appropriate parameter to zero.

The two energy functions can be applied to dense data and with small modification to sparse data as well. The problem of surface reconstruction and retrieving images from sparse data is addressed and an algorithm to perform these tasks is obtained and implemented.

The paper is organized in the following way: Section 2 presents an overview of Markov Random Fields in vision. Section 3 propose the deterministic approximation of MRF, with an application to the two particular models mentioned above. Section 4 discusses how to estimate the parameters of the models. In Section 5 the results are exhibited. Section 6 Extends the formulation to sparse data and show results. Section 7 concludes the paper.

2 MRF for Surface Reconstruction

Consider the problem of approximating a surface given a set of sparse and noisy data g on a regular 2D lattice. We think the surface as a field f (surface-field) defined on a regular lattice, such that the value of this field at each site of the lattice is given by the surface value at this site. Adopting a probabilistic point of view we are interested in the conditional probability of f given the data g ($P(f|g)$). Bayes theorem allows us to write

$$P(f|g) \propto P(g|f)P(f) \tag{2.1}$$

where $P(g|f)$ is essentially the probability distribution of the noise and $P(f)$ is the prior probability distribution of the field f. The noise is usually assumed to be Gaussian, so that $P(g|f)$ is known. The shape of $P(f)$ depends on our a priori information about the system and it is what differentiates a model from an other one. If we assume that the probability of a certain value of the field at a site depends only upon the neighbor sites, the Clifford-Hammersley theorem guarantees that we can always write

$$P(f) \propto e^{-\beta U(f)} \tag{2.1}$$

where $U(f)$ can be computed as the sum of local contributions from each lattice site i and β is a parameter that is called the inverse of the natural temperature of the field. As a result the conditional probability $P(f|g)$ can always be written as $P(f|g) \propto \frac{1}{Z}e^{-\beta H_g(f)}$ where $H_g(f)$ is usually called the "energy function" of the model.

The discontinuities of the field f can easily be included in this framework. In fact Geman and Geman [8] introduced the idea of another field, the line process l, located on the dual lattice, and representing explicitly the presence or absence of discontinuities that break the smoothness assumption. The interaction between the fields f and l can be chosen so that the most likely configurations are piecewise smooth.

Once the probability distribution has been written down, an estimate for the fields is usually obtained with the field values that maximizes it, or equivalently that minimize the energy function $H_g(f)$. A number of troubles immediately arises. In fact the energy function very often is not convex, and due to the discrete nature of the line process fields, simulated annealing or similar Monte Carlo techniques must be used to solve the problem. The computational effort to obtain a good estimate of the fields is then very large, becoming one of the major drawbacks of the MRFs. Another drawback comes from the fact that the energy function depends on some parameters that control the relative weight of the various terms. The problem of parameter estimation has been attacked in many ways, but it is far to be completely solved. It is still not very clear how they are related to the quality of the solution and to quantities of physical interest.

3 A deterministic approximation of MRF

3.1 The Effective Potential and the Deterministic Equations

We now briefly sketch how a deterministic approach to MRF can be pursued. Details of the derivation can be found in [6]. Let g a given set of data, possibly sparse, defined on a $2D$ lattice, f the field associated to the surface to reconstruct and l a field whose value is one where a discontinuity occurs and zero elsewhere. We consider energy functions of the general form

$$H_g[f,l] = E_{fg}[f,g] + E_{fl}[f,l] \qquad (3.1.1)$$

where the first term is usually $\sum_i (f_i - g_i)^2$, coming out from the Gaussian distribution of the noise, and the other terms contain the a priori information about the system. Due to the discrete nature of the line process field the minimum of the energy function 3.1 can not be found by computing derivatives with respect to the variables, unless we succeed in eliminating the line process field from the probability distribution. This can be done by looking at the partition function Z. It is well known that Z completely describes the statistical properties of a system, and its knowledge permits to compute all the relevant statistical quantities (as mean values or correlation functions). In our case it can be written in the following form:

$$Z = \sum_{\{f,l\}} e^{-\beta H_g[f,l]} = \sum_{\{f\}} e^{-\beta(E_{fg}[f,g])} \sum_{\{l\}} e^{-\beta E_{fl}[f,l]} \; .$$

where $\sum_{\{f,l\}}$ means the sum over all the possible configurations of the fields f and l.Defining a temperature dependent *effective potential* as

$$E^{\beta}_{eff}[f] = -\frac{1}{\beta} \ln \sum_{\{l\}} e^{-\beta E_{fl}[f,l]}$$

the partition function becomes:

$$Z = \sum_{\{f\}} e^{-\beta(E_{fg}[f,g] + E^{\beta}_{eff}[f])} \qquad (3.1.2) \; .$$

We notice that the line process disappeared from equation 3.1.2, leaving Z unchanged, so that the energy function given by $E_{fg} + E^{\beta}_{eff}$ has the same information content of H_g. Information coming from the discontinuity field is contained in the effective potential, who "simulates" the presence of the line process. A set of deterministic equation is now easily obtained by simply minimizing the new energy function $E_{fg} + E^{\beta}_{eff}$ with respect to f, that is by writing down the following system of equations (usually non linear):

$$\frac{\partial}{\partial f_i}(E_{fg} + E^{\beta}_{eff}) = 0 \qquad (3.1.3) \; .$$

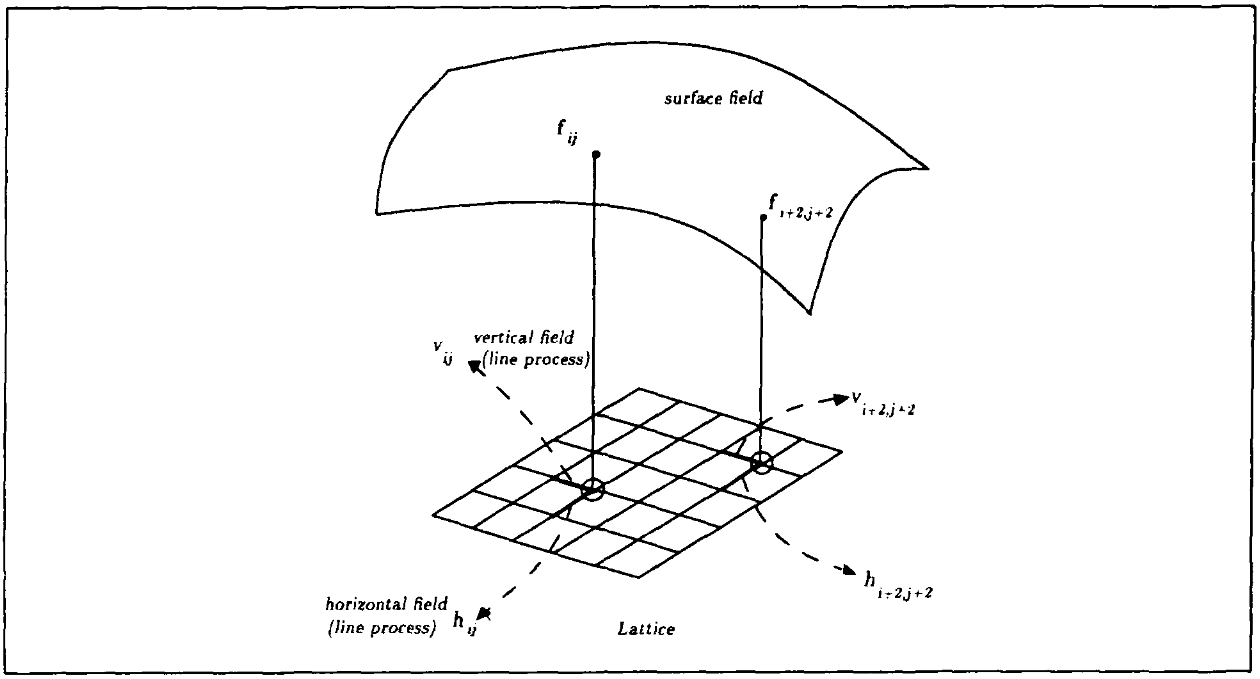


Figure 1: *The surface field f, the horizontal line process h and the vertical line process v are represented in two sites, (i,j) and (i+2, j+2), of the lattice. There are three fields defined in each site of the lattice as opposed to having one field in a lattice and other fields in a dual lattice.*

If the temperature $(\frac{1}{\beta})$ is different from zero eq. 3.1.3 gives implicitely the mean statistical values of the field f in the so called *mean field approximation.* When the temperature is zero it can be shown that eq. 3.1.3 gives the values of f that minimizes $H_g[f, l]$.

Similar equation can be derived for the mean values of the discontinuity field. In this case the trick is to add the term $\sum_i l_i h_i$ to the energy function, where h is an arbitrary external field that is set to zero at the end of the calculations. The effective potential depend then on the field too, but it can be shown that in the mean field approximation the following equation hold:

$$< l_i >= \frac{\partial}{\partial h_i} E^{\beta}_{eff} \Big|_{h=0} \qquad (3.1.4)$$

Equation 3.1.4 becomes exact in the zero temperature limit, and together eq. 3.1.3 gives a set of deterministic equations whose inspection is generally very useful, giving insights on the nature of the solution and of the parameter of the model.

This method is very general and can be applied, in principle, to every energy function tha can be written as in eq. 3.1. even if its appliability depends on how far the analytical computations can be carried on. We showed here its basic formulation including a line process field, but it can obvioulsy extended. This turns out to be useful, because in this paper we introduce two discontinuity fields (see figure 1) : the horizontal (h) and vertical (v) line process. The line process h_{ij} connects the site (i, j) to the site $(i, j-1)$, while v_{ij} connects the site (i, j) to the site $(i-1, j)$. In this way we will be able to reduce the two dimensional problem to two one dimensional problems, provided that the horizontal and vertical line processes do not interact. In the next section we consider some specific energy functions and list some exact and approximated result that have been obtained with these techniques in [6].

3.2 A MRF Model for Smoothing and Detecting Discontinuities

In this section we study a particular MRF model, defined by the following energy function:

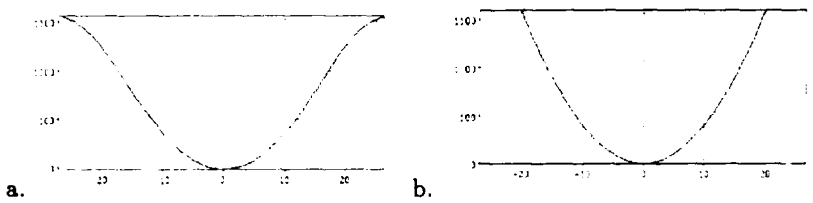


Figure 2: *The effective potential is shown as a function of the horizontal gradient* $(\bar{f}_{i,j} - \bar{f}_{i-1,j})$ *a) For* $\beta = 0.002$. *b) Zero temperature limit* $(\beta \rightarrow \infty)$.

$$E(f, h, v) = E_{fg}(f) + E_l(h, v) + E_{fl}(f, h, v) \tag{3.2.1}$$

where

$$E_{fg}(f) = \sum_{i,j} (f_{i,j} - g_{i,j})^2 \quad , \quad E_l(h, v) = \gamma \sum_{i,j} (h_{i,j} + v_{i,j}) \tag{3.2.1a}$$

$$E_{fl}(f, h, v) = \alpha \sum_{i,j} [(f_{i,j} - f_{i,j-1})^2(1 - h_{i,j}) + (f_{i,j} - f_{i-1,j})^2(1 - v_{i,j})] \tag{3.2.1b}$$

where α and γ are positive valued parameters and g is the data field. This energy function has been already studied in the context of surface approximation,[1][10] [12][11] but within different approaches. Let us briefly comment each term of eq. 3.2.1. The first term enforces closeness to the data and the second one takes into account the price we pay each time we create a discontinuity and is necessary to prevent the creation of discontinuities everywhere. The third term contains the interaction between the field and the line processes: if the horizontal or vertical gradient is very high at site (i, j) the corresponding line process will be very likely to be active ($h_{i,j} = 1$ or $v_{i,j} = 1$), to make energy decrease and signal a discontinuity.

We notice that the line process fields interacts only with the f field: there is not interaction between the line process fields themselves. Therefore the contribution of the line process to the partition function can be exactly computed and the effective potential turns out to be:

$$\begin{aligned} E^{\beta}_{eff}(f) = \quad & \textstyle\sum_{i,j} \{\alpha((\bar{f}_{i,j} - \bar{f}_{i-1,j})^2 + (\bar{f}_{i,j} - \bar{f}_{i,j-1})^2) + \\ & \tfrac{1}{\beta} ln[\sigma_\beta(\gamma - \alpha(\bar{f}_{i,j} - \bar{f}_{i-1,j})^2)\sigma_\beta(\gamma - \alpha(\bar{f}_{i,j} - \bar{f}_{i,j-1})^2)]\} \quad (3.2.2) \end{aligned}$$

where we have defined $\sigma_\beta(x) = \frac{1}{1+e^{-\beta x}}$.
The effective potential "simulates" the interaction between the line process and the field f with a new interaction between the field and itself. We notice from eq. 3.2.2 that the effective potential is a sum of local terms depending on the gradient of the field and on the temperature of the system. The shape of one of the local terms has been depicted in fig. 2 for $\beta = \infty$ and $\beta = 0.002$ as a function of the horizontal gradient of the field.

If no line process is present in the energy function its shape would be a simple parabola, increasing to infinity as the gradient increase to enforce smoothness. Let us consider the zero temperature result of fig. 2: as far as the gradient is below a threshold the effective potential is a parabola, then it becomes a constant. This means that the smoothing effect acts only where the gradient is not too large. The gradient growing too much means that a discontinuity is likely to occur, and no smoothing effect has to take place (the potential becomes constant). When the temperature is different from zero the interpretation is similar, but the border between the smoothing and not smoothing regions becomes less sharp, due to thermal fluctuations of the line process field. This intuition is supported by an explicit computation: by applying eq. 3.1.4 we obtain:

$$\bar{h}_{i,j} = \sigma_\beta(\alpha(\bar{f}_{i,j} - \bar{f}_{i-1,j})^2 - \gamma) \quad and \quad \bar{v}_{i,j} = \sigma_\beta(\alpha(\bar{f}_{i,j} - \bar{f}_{i,j-1})^2 - \gamma) \tag{3.2.3}.$$

Equations 3.2.3 allow us to compute the mean values of the line process as a function of the mean values of the field f. In the zero temperature limit ($\beta \rightarrow \infty$) the function σ_β becomes the Heaviside function: when the horizontal or vertical gradient ($\bar{f}_{i,j} - \bar{f}_{i,j-1}$ or $\bar{f}_{i,j} - \bar{f}_{i-1,j}$) are larger than a threshold ($\sqrt{\frac{\gamma}{\alpha}}$) a vertical or horizontal discontinuity is created, since the price to smooth the function at that site is too high. This leads to a clear interpretation of the parameter γ, as it will be discussed in section 4.2.

The values of the field appearing in eq 3.2.3 can be found by solving the set of deterministic equation described in the previous section (eq. 3.1.3), that can be written as:

$$\begin{aligned}\bar{f}_{i,j} = \quad & g_{i,j} - \alpha(\bar{f}_{i,j} - \bar{f}_{i,j-1})(1 - \bar{v}_{i,j}) + \quad \alpha(\bar{f}_{i,j+1} - \bar{f}_{i,j})(1 - \bar{v}_{i,j+1}) \\ & - \quad \alpha(\bar{f}_{i,j} - \bar{f}_{i-1,j})(1 - \bar{h}_{i,j}) + \quad \alpha(\bar{f}_{i+1,j} - \bar{f}_{i,j})(1 - \bar{h}_{i+1,j}) \qquad (3.2.4)\end{aligned}$$

where $\bar{h}_{i,j}$ and $\bar{v}_{i,j}$ are given by eq. 3.2.3

When substituting eq. 3.2.3 in eq. 3.2.4 we obtain a set of non linear equations for the mean values of f, that can be solved in a fast, iterative and parallel way. Equation 3.2.4 gives the field at site i, j as the sum of data at the same site, plus an average of the field at its neighbor sites. This average takes in account the difference between the neighbors. The larger is the difference, the smaller is the contribution to the average. This is captured by the term $(1 - l_{i,j})$, where $l_{i,j}$ is the line process. At the zero temperature limit ($\beta \rightarrow \infty$) the line process becomes 1 or 0 and then only terms smaller than a threshold must be taken in account for the average. This interpretation helps us in understanding the role of the α and γ parameters, as it will be discussed in chapter 4.

3.2.1 The Effective Potential and the Graduate Non Convexity Algorithm

We have to point out that this energy function has been studied by Blake and Zisserman [1], in the context of edge detection and surface interpolation. They do not derive the results from the MRF formulation but they simply minimize the energy function. From a statistical mechanics point of view the mean-field solution does not minimize the energy function, but this becomes true in the zero temperature limit, so their approach must be recovered from the MRF formulation in this limit. This is indeed the case, and it is easy to show that the effective potential becomes the Blake and Zisserman potential when β goes to infinity. In order to obtain the minimum of the energy function E_2 Blake and Zisserman introduce the GNC (Graduate Non Convexity) algorithm, that is different from our deterministic scheme, but can be embedded in the MRF framework in a natural way. Let us review briefly the GNC algorithm. The main problem with the this is that is not a convex function and a gradient descent method can not be applied to obtain the minimum because one could be trapped in a local minimum. In order to solve this problem Blake and Zisserman introduce a family of energy functions $E^{(p)}$, depending continuously on a parameter p, $p \epsilon [0, 1]$, such that $E^{(1)}$ is convex, $E^{(0)} \equiv E_2$ and $E^{(p)}$ are non convex for $p \epsilon [0, 1)$. Gradient descent is successively applied to the energy function $E^{(p)}$ for a prescribed decreasing sequence of values of p starting from $p = 1$, and this procedure is proved to converge for a class of given data. The construction of the family of energy functions $E^{(p)}$is *ad hoc* and consists of a piecewise polynomial. In our framework a family of energy functions with such properties is naturally given by ${E_{eff}}^{\beta}$. The GNC algorithm can then be interpreted as the tracking of the minimum of the energy function as the temperature is lowered to zero (like a deterministic annealing). In this way the approach of Blake and Zisserman can be view as a deterministic solution of the MRF problem, even if it does not fully exploit the possibility of obtaining deterministic equations for the surface and discontinuity fields. The results obtained by applying this method to edge detection, for example, are good, and a pattern of meaningful discontinuities can usually be recovered [1]. However, sometimes the full set of discontinuities is not obtained: when the gradient of the image brightness is under the threshold a discontinuity may not be detected, even though it would be necessary, for example, to close a contour.

If interaction between lines (self interaction of the line-field) is introduced in the energy function this problem can be overcome, as we discuss next.

3.3 Making discontinuities smooth

So far we have not exploited an important physical constraint of images, namely the smoothness of the discontinuity field. Isolated discontinuities are very unlikely to occur and, on the contrary, the presence of a discontinuity at a site makes more likely the presence of a discontinuity at a neighboring site. This

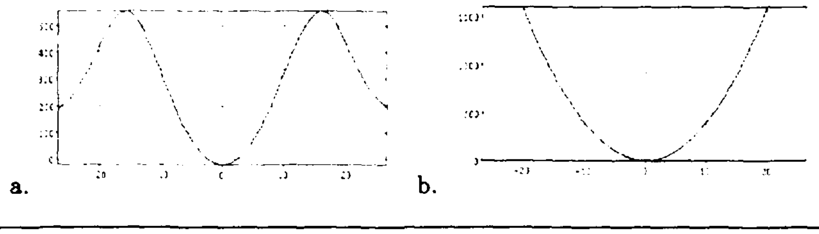


Figure 3: *The effective potential for $\beta = 0.002$ and different values for the parameter ϵ. a) For $\epsilon = 0.9$ b) For $\epsilon = 0$ the effective potential becomes the potential studied by Blake and Zisserman.*

smoothness constraint on the discontinuity field can be incorporated in the model by simply adding a new term to the energy function. We then study a MRF model defined by the following energy function

$$E' = E + E_{ll}$$

where E is given by equation 3.2.1. We have defined the new term as

$$E_{ll} = -\epsilon\gamma \sum_{i,i} [h_{i,j} h_{i,j-1} + v_{i,j} v_{i-1,j}]$$

where ϵ is a new parameter whose exact meaning and estimation will be explained in the next section. To make more evident the meaning of the new term we notice that

$$E_l + E_{ll} = \sum_{ij} \gamma(1-\epsilon)[h_{i,j} + v_{i,j}] + \frac{1}{2}\epsilon\gamma[(h_{i,j} - h_{i,j-1})^2 + (v_{i,j} - v_{i-1,j})^2] \qquad (3.3.1)$$

where E_l is given by equation 3.2.1a. From eq. (3.3.1) it is clear that ϵ is related to the degree of smoothness of the discontinuity field and so must be a positive number. In order to keep positive the price we pay for creating a discontinuity and to prevent the line processes from being active everywhere, ϵ has to be less than 1.

We notice here that this model is a simplified version of other potentials. In particular, the neighbor size considered here is at most of two pixels; Gamble & Poggio for example, have discussed more sophisticated cliques composed of larger neighboors.

As in the previous case we are interested in computing the contribution of the line process to the partition function. This task is more difficult than the previous one, and we used the mean field approximation to obtain approximated result for the effective potential.

$$\begin{aligned} E_{eff}(\bar{f}) = \; & \textstyle\sum_{i,j} \{\alpha((\bar{f}_{i,j} - \bar{f}_{i-1,j})^2 + (\bar{f}_{i,j} - \bar{f}_{i,j-1})^2) \\ & -\tfrac{1}{\beta} ln[(1 + e^{-\beta(G^h_{i,j} - \epsilon\gamma \frac{\bar{h}_{i,j-1} + \bar{h}_{i,j+1}}{2})})(1 + e^{-\beta(G^v_{i,j} - \epsilon\gamma \frac{\bar{v}_{i-1,j} + \bar{v}_{i+1,j}}{2})})]\} \qquad (3.3.2) \end{aligned}$$

where $G^h_{i,j} = \gamma - \alpha(\bar{f}_{i,j} - \bar{f}_{i-1,j})^2$ and $G^v_{i,j}$ is analogous. A detailed computation by means of the transfer matrix method shows that this potential is highly non local, due to the interaction between the discontinuity fields. To have some insight on the effect of the new term in the energy function we have looked for a *local approximation* to the non local effective potential. To obtain a local potential we approximate the term $\frac{\bar{h}_{i,j-1} + \bar{h}_{i,j+1}}{2}$ by $\bar{h}_{ij}$. The result obtained is plotted in fig. 3 as a function of the horizontal gradient.

The effect of this new effective potential can be well understood if we think at the system as an ensamble of interacting particles and if we study the interaction force between particles, that is (minus) the derivative of the effective potential. In this case the gradient of the field should be thought as the relative distance between to particles. The force has been depicted in fig. 4 for two values of ϵ and for $\beta = 0.002$. We notice that when the gradient is low the force is linear and attractive, as the force of the ideal spring. When the gradient increases the force quickly decreases, and, unlike the usual spring, becomes repulsive, pushing the particles apart. This effect takes places only in a limited interval of values of the gradient; when it becomes too large the spring breaks up and the force goes to zero. As a result the overall effect will be of a smoothing,

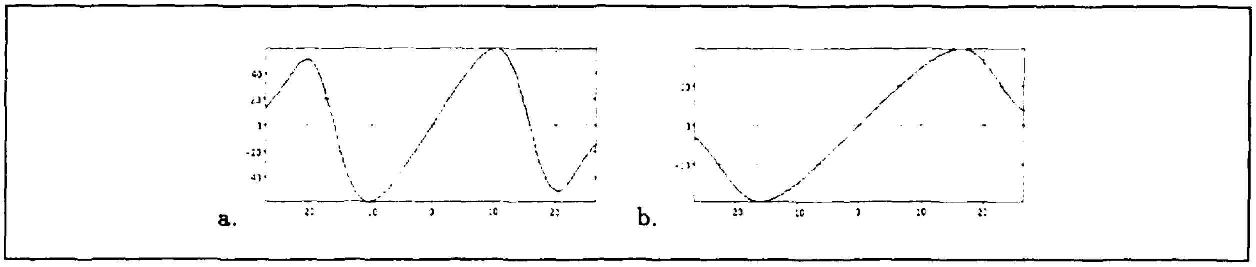


Figure 4: *The force associated to the effective potential for $\beta = 0.002$ and different values of the parameter ϵ a)For $\epsilon=0.9$. b) For $\epsilon = 0$ the force does not assume negative values, and the enhancing effect does not take place.*

where the gradient is smaller than a threshold, and an *enhancing*, where the gradient is "sufficiently large". Where the gradient is too large no smoothing or enhancing will take place.
We notice that the "enhancing" effect is due to the new term E_{ll} in the energy function, and its intensity is controlled by the parameter ϵ. By setting ϵ to zero we recover the previous, and the force associated to the effective potential never becomes negative (fig. 4b).
This result could have been obtained just by qualitative reasoning. In fact the term E_{ll} has been introduced to make the presence of a discontinuity at a site to stimulate the presence of another discontinuity at a neighbor site: since a discontinuity is created where the gradient is large, to obtain this effect it is needed that a large gradient at a site makes the gradient at its neighbor site larger, and this is what the effective potential of fig. 3 does.
As in the previous case we obtained a set of non linear equation relating the mean values of the discontinuity field with the mean values of the field:

$$\bar{h}_{i,j} = \sigma_\beta(\alpha(\bar{f}_{i,j} - \bar{f}_{i,j-1})^2 - \gamma + \epsilon\gamma\frac{\bar{h}_{i,j-1} + \bar{h}_{i,j+1}}{2}) \quad \text{and}$$

$$\bar{v}_{i,j} = \sigma_\beta(\alpha(\bar{f}_{i,j} - \bar{f}_{i-1,j})^2 - \gamma + \epsilon\gamma\frac{\bar{v}_{i-1,j} + \bar{v}_{i+1,j}}{2}) \qquad (3.3.3)$$

Notice that now equations 3.14 form a set of non linear equations and the solution for the line process is not as simple as before.

We applied the arguments of the previous section to obtain a deterministic equation for the field f. We do not write it here for sake of space, (they can be found in [6]) and simply remind that they are a specific instance of eq. 3.3. We used for E'_{eff} the formula coming out of our computations, but every potential reproducing the shape of fig. 3 should work.

3.3.1 Hysteresis, Suprathreshold and Threshold

The solution obtained in equation 3.3.3 for the line process deals with the problem of streaking. Streaking is the breaking up of an edge contour caused by fluctuations above and below a threshold along a contour [2]. This is a common problem with thresholded detectors. In the results coming out from the last MRF model we considered the thresholding is done with hysteresis. This is the way the Canny's detector works [2]. The hysteresis phenomena are evident from equation 3.3.3, since, with the creation of a horizontal line at a neighbor site, say $\{i, j-1\}$ $(h_{i,j-1} = 1)$, the energy necessary for creating a line at a site $\{i, j\}$ decreases by $\frac{\epsilon\gamma}{2}$, if two lines are created at $\{i, j-1\}$ and $\{i, j+1\}$ then the energy decreases by $\epsilon\gamma$. A low threshold (threshold) and a high threshold (suprathreshold) arise naturally. The suprathreshold for creating a line is given by $\frac{\gamma}{\alpha} \leq (\bar{f}_{i,j} - \bar{f}_{i,j-1})^2$, in this case a line is created no matter what. In the same way the lowest threshold is given by $\frac{\gamma}{\alpha}(1-\epsilon) \leq (\bar{f}_{i,j} - \bar{f}_{i-1,j})^2 \leq \frac{\gamma}{\alpha}$, in this case a horizontal line will be created at site $\{i, j\}$ if a horizontal line has been created in the sites $\{i, j-1\}$ and $\{i, j+1\}$.
At higher temperatures the threshold and suprathreshold are not so well - defined, and we have adaptative thresholds that depend on the values of the field.

4 Parameter Estimation

The parameters α, γ and ϵ must be estimated in order to develop an algorithm that smoothes, enhances and finds the discontinuities a given set of data.

4.1 The parameter α

To estimate α we notice, *a posteriori*, that if the gradient is very small the discontinuity field has no effect and can be neglected. We analyze α in this case, assuming that it does not change at higher energies.
The MRF model obtained neglecting discontinuities has been shown to be equivalent to standard regularization. The parameter α plays then the role of the *regularization parameter*, controlling the balance between the data and the smoothing term. The noisier are the data the less we want to "trust" them, so that α is large; the less noisy are the data the more we "trust" them, so that α should be small. To estimate α various mathematical methods are available. The generalized cross validation method introduced by Wahba [15, ?] and the standard regularization method described by Tikhonov [14] were studied by Geiger and Poggio [7] to estimate α.
The generalized cross validation method (GCV) is used when an estimation of the noise is not available. It states that the optimal value of α can be obtained by minimizing the functional (here in one dimension)

$$V(\alpha) = \frac{1}{n} \sum_{i=0}^{n} \frac{[f_{n,\alpha}(t_i) - g_i]^2}{(1 - a_{kk}(\alpha))^2} \omega_k^2(\alpha)$$

where $f_{n,\alpha}(t_i)$ is the smoothed solution, $\omega_k(\alpha) = (1 - a_{kk}(\alpha))/(1 - \frac{1}{n}\sum_{j=0}^{n} a_{jj}(\alpha))$ and $a_{kk}(\alpha) = \frac{\theta}{\theta f_k}(f_{n,\alpha})(t_k)$.
For the purpose of smoothing images GCV and standard regularization methods gives about the same values of α [7]. Better results should be obtained if the estimation of α is done locally (for a small neighborhood).

4.2 The parameter γ

We first analyze the parameter γ setting ϵ, to zero. From the line process equation given by equation 3.3 one can see that $\sqrt{\frac{\gamma}{\alpha}}$ is the threshold for creating a line. Let's call ξ the value $\sqrt{\frac{\gamma}{\alpha}}$. From the expression of the effective potential we notice that if the gradient is above ξ no smoothing is done and if the gradient is below ξ then smoothing is applied. The parameter ξ defines the resolution of the system and we explain this with two examples.
In the first example suppose we are working with the stereo module, so the data field is a depth-field. In this case ξ is the threshold for changes in depth to be called a depth discontinuity. This value is determined according to the resolution of the stereo system available and/or to the desired resolution that one is interested.
If the field is associated to intensity data the parameter ξ is the threshold for detecting edges. This value is somehow arbitrary, and probably context dependent. A variation of $\xi = 16$ units to $\xi = 32$ units for a 8-bit array image is likely to be in agreement with the human threshold. The exact value of ξ depends on the attention of the observer or on the sensitivity required by the examined problem.

For a value of ϵ diferent from zero the absolute threshold to create an edge is also given by ξ. However when there is support from neighbor edges this threshold can be lowered to $\xi \times (\sqrt{(1 - \epsilon)}$ (see equation 3.3.3). This suggests to set γ so as to guarantee that the highest noise gap is smoothed (we are assuming that noise gaps are isolated features). In this case the ϵ parameter will be set to assure that at the edges the images will not be smoothed but perhaps enhanced. In this case the value of $\xi = 30$ may be desired for an 8-bit array.

4.3 The parameter ϵ

The parameter ϵ makes the energy E' different and more general than the previous one. It controls the amount of propagation of the line in a line process. So, once a line is created, the price to pay for creating another line next to it will be lowered by the amount of $\gamma\epsilon$. In other words, from the definition of E' one can see that the difference in the energy corresponding to the creation or not of a line at pixel $(i - 1, j)$ is given by $\gamma\epsilon$. This is what characterizes the threshold and suprathreshold or the hysteresis phenomena [2].

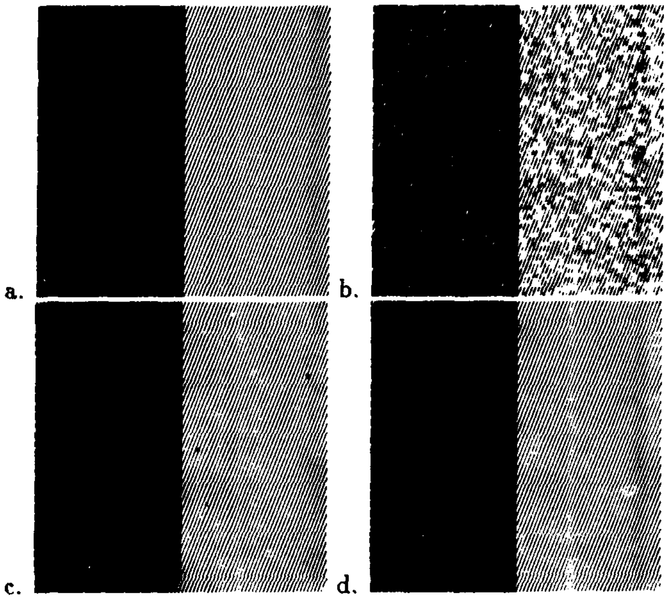


Figure 5: *a) Step edge with 140 grey value units for the step. 1b) White noise with standard deviation 30 grey value units has been added. c) The noisy image after 20 iterations for* $\alpha = 4$, $\gamma = 24000$, $\epsilon = 0$. *d) The noisy images after 20 iteration for* $\alpha = 4$, $\gamma = 45000$, $\epsilon = 0.9$

The threshold is given by $\sqrt{\frac{\gamma(1-\epsilon)}{\alpha}}$ and the suprathreshold by $\sqrt{\frac{T}{\alpha}}$, ϵ varying from 0 to 1.0. When $\epsilon =$ 1.0 lines are created everywhere, since once a line is created there is no cost in creating another one and then it propagates indefinitely. How much does one want to propagate a line? or How much should the difference between the threshold and the suprathreshold be? or Which exact value of ϵ to choose? According to the discussion above about γ we conclude that ϵ should be chosen to guarantee that $\sqrt{\frac{\gamma(1-\epsilon)}{\alpha}}$ is below the desired edge threshold. For detecting edges of objects in a scene one wants to have high suprathreshold and threshold (usually object boundaries exhibit high gradients) and ϵ large (bigger than 0.5) so that all the object boundaries are detected, included the exceptional boundary pixels with a somehow smaller gradient.

5 Results

For the implementation the zero temperature limit equations have provided results as good as the deterministic annealing with a faster computational time. We do not have proves of convergence but only good experimental results.

In order to find the mean field solution for the model described by the energy function E' we solved the deterministic equations for the line process and the field f in a coupled and iterative way. Details about implentation can be found in [6] Typically the algorithm has converged in 20 iterations which takes about 1 minute for images of 64 X 64 pixels on a Symbolics 3600.

We first tested the algorithm on a synthetic step edge image with noise added. This is a good test-image since locally many edges on real images are like that. Due to the simplicity of the pattern the effect of having added a new term to the energy function E is very clear.

The step edge image is an 8-bit array of 64 × 64 pixels with a step intensity of 140 units (see Figure 5a). Next white noise, with standard deviation 30, is added to the step edge (see fig. 5b). We used the deterministic equation derived from the energy functions E and E' to obtain fig. 5c and d. Fig. 5c has been obtained after 20 iterations of the iterative algorithm, setting $\alpha = 4$ and $\gamma = 24000$ so that $\sqrt{\frac{T}{\alpha}} \approx 80$. Notice that the noise has not been completely eliminated, and the edge starts to be smoothed. To completely eliminate

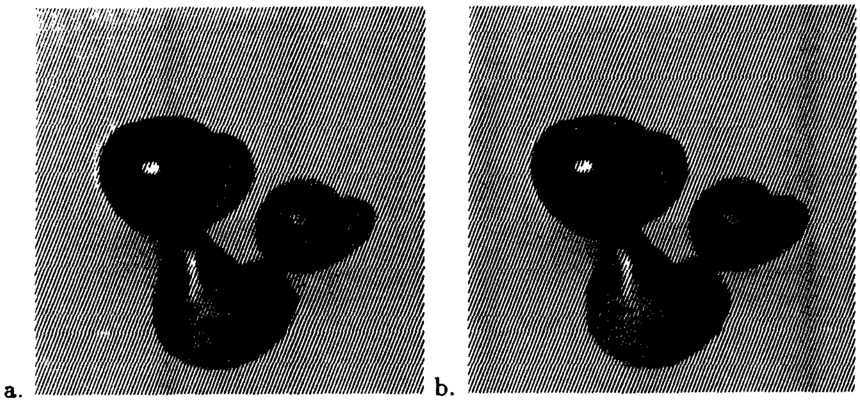


Figure 6: *a) The still life image 128 X 128 pixels. b) The image smoothed with* $\epsilon = 0.9$, $\gamma = 1400$ *and* $\alpha = 4$ *for 20 iterations*

the noise we could choose a larger value of γ, enlarging the smoothing region, but as a result the step edge would be smoothed too.
In fig. 5d we set $\epsilon = 0.9$, $\alpha = 4$ and $\gamma = 45000$. Because we set the ratio $\frac{\gamma}{\alpha}$ to be higher all the noise has been smoothed away without smoothing the edge. Indeed at the edge there is support from the neighbor edges to lower the threshold to a factor $(1 - \epsilon) = 0.1$. Therefore the noise is supressed and the edges are not smoothed but on the contrary are enhanced. We conclude here that for a noisy step edge, the model described by the energy function E' is able to retrieve and enhance the edge in a better way than the model associated to E.

We used the same algorithm used to obtain fig. 5d to analyze a real still life image. The original image has been shown in fig. 6a, and the smoothed one in fig. 6b. It can be seen that specular, shadow and contour edges have been enhanced, while the noise has been smoothed away.

6 Sparse Data

Until now we dealt with a set of data defined on all the lattice. This is not always the case, especially if the field has to be the output of a stereo or motion module. In that case data are given on a subset of the lattice and the data-field term (E_{fg}) in the energy function can be modified in the following way

$$E_{fg} = \sum_{i,j}(f_{i,j} - g_{i,j})^2 \Rightarrow \sum_{i,j}(f_{i,j} - g_{i,j})^2\gamma_{i,j}.$$

Here $\gamma_{i,j}$ is a flag that is one if data is given at site (i,j) and 0 otherwise. This slightly modification has no effect on the theoretical results, and only some changes take place in the deterministic equations for the surface field: the term enforcing closeness to data disappears when data is not given at a site. We rewrite here the deterministic solution for the field f in the case of the Energy E, since the energy equations for E' are modified in the same way. In the case of sparse data eq. 3.2.4 becomes then

$$\begin{aligned}\bar{f}_{i,j}\gamma_{i,j} = {} & g_{i,j}\gamma_{i,j} - \alpha(\bar{f}_{i,j} - \bar{f}_{i,j-1})(1 - \bar{v}_{i,j}) + \alpha(\bar{f}_{i,j+1} - \bar{f}_{i,j})(1 - \bar{v}_{i,j+1}) \\ & - \alpha(\bar{f}_{i,j} - \bar{f}_{i-1,j})(1 - \bar{h}_{i,j}) + \alpha(\bar{f}_{i+1,j} - \bar{f}_{i,j})(1 - \bar{h}_{i+1,j})\end{aligned}$$

To apply this algorithm one needs to fill in the data. We chose to fill in the data by averaging near neighboors and applying the algorithm at the same time. So at each step of the algorithm each lattice site is visited: if there is no field value the average neighbor value is taken otherwise we apply the above algorithm. We notice that no action is taken if at a particular site there is no field value and no neighbor field value.

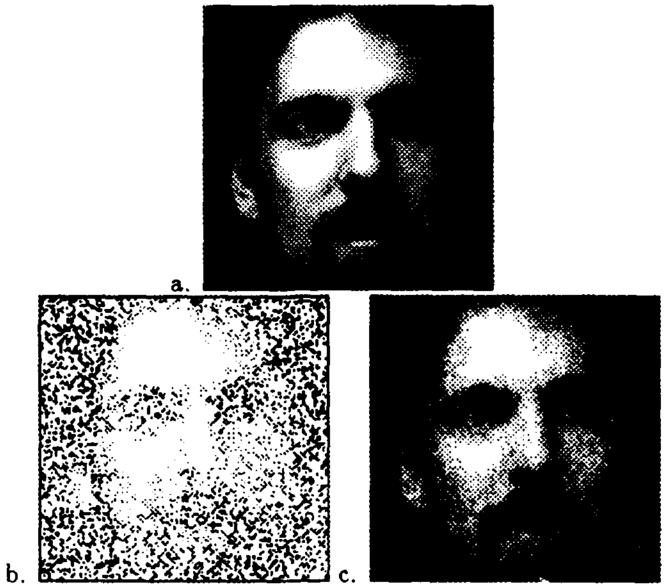


Figure 7: *a) A face image of 8-bits and 128 X 128 pixels. b) Randomly choosen 70 % of the original image. For display the other 30% are filled with white dots. c) The algorithm described above is applied to smooth and fill in at the same time with* $\epsilon = 0.9$, $\gamma = 1400$ *and* $\alpha = 4$ *for 70 iterations.*

From one face image we produced sparce data by randomly suppressing 70 % of the data. (see Figure 7). We then applied the third energy algorithm to sparse data. The parameters were kept the same as the other real image.

The reconstruction of images from sparse data can be applied to depth data at zero crossings in which is known as surface reconstruction. We used the stereo algorithm based on zero crossings to obtain depth data from the image shown below. Then we applied susscesfully the algorithm to reconstruct the depth surface. The parameters now have to change according to the criteria used for depth discontinuity.

7 Conclusion

We have used statistical mechanics tools to derive deterministic approximations of Markov Random Fields models. In particular we studied a model that is suitable for surface reconstruction preserving discontinuities. This model has been developed to include the following characteristics:

- the surface field is smoothed when its gradient is not too high,
- where a discontinuity occurs contrast will be enhanced (if it is not too large already),
- the discontinuity field is likely to be smooth (isolated discontinuities are inhibited),
- hysteresis and adaptative multiple threshold arise naturally from the model.
- 3 parameters are needed to specify the model,
- An understanding of the parameters is possible.
- It is naturally absorbed in the model the problem of sparse data

We also derived a deterministic solution for the mean values of the surface and discontinuity fields, consisting in a system of coupled non linear equations. An algorithm has been implemented to obtain a

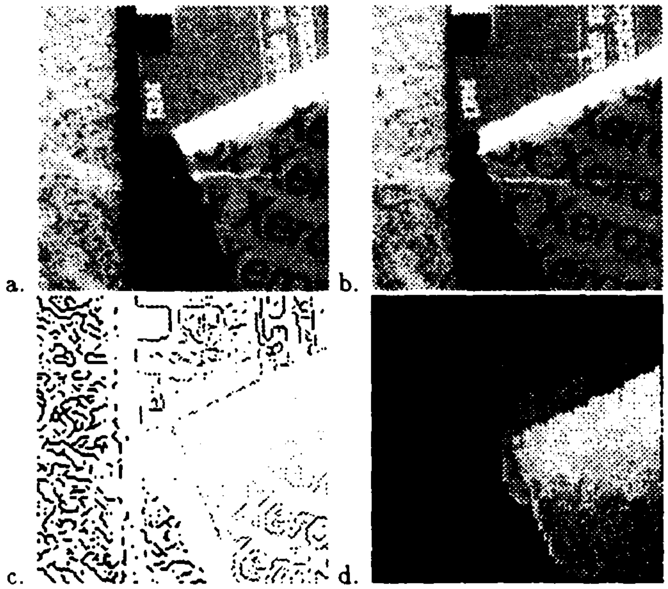


Figure 8: *a) A left camera image of 8-bits and 128 X 128 pixels. b) The right camera of the same scene c) Sparse depth data obtained by the stereo algorithm based on zero crossing. The intensity represents depth values. d) The algorithm is applied with* $\epsilon = 0.0$, $\gamma = 400$ *and* $\alpha = 4$ *and the surface reconstruction is obtained.*

solution for this system: it is fully parallelizable, iterative and recursive, allowing low computational time. It can be applied to sparse data. Some approximations have been done to obtain local results; it would be interesting to analyze other approximations or extensions of this model. For example, a model that includes interaction between the horizontal and vertical line processes could be developed to inhibit self-intersections of the discontinuity field. A term like $h_{ij}(1 - v_{ij})$ would do it.

The integration of different visual modules to improve the detection of the discontinuities can also be addressed in this scheme. For instance, as suggested by Gamble & Poggio [5], we can add the term $\delta\, h_{ij}(1-e_{ij})$ to the Weak Membrane Energy or the third energy. Here e_{ij} is an external field, for example the edge map that is coupled with the stereo field. For implementation purposes the only consequence of adding this term is the change of the global parameter γ into the local parameter $\gamma'_{ij} = \gamma - \delta(1 - e_{ij})$. For a more precise discussion see [6]. Another deterministic scheme is presented by Hurlbert and Poggio [9].

Acknowledgments

This paper describes research done within the Center for Biological Information Processing, in the Department of Brain and Cognitive Sciences, and at the Artificial Intelligence Laboratory. This research is sponsored by a grant from the Office of Naval Research (ONR), Cognitive and Neural Sciences Division; by the Artificial Intelligence Center of Hughes Aircraft Corporation; by the Alfred P. Sloan Foundation; by the National Science Foundation; by the Artificial Intelligence Center of Hughes Aircraft Corporation (S1-801534-2); and by the NATO Scientific Affairs Division (0403/87). Support for the A. I. Laboratory's artificial intelligence research is provided by the Advanced Research Projects Agency of the Department of Defense under Army contract DACA76-85-C-0010, and in part by ONR contract N00014-85-K-0124.

We are grateful to Tomaso Poggio for his guidance, suggestions and support. Eric Grimson and Woody Yang read early drafts and provided us helpful comments, Ed Gamble and Alessandro Verri also brought to us useful suggestions.

References

[1] Andrew Blake and Andrew Zisserman. *Visual Reconstruction*. MIT Press, Cambridge, Mass, 1987.

[2] John F. Canny. Finding lines and edges in images. Technical Report TM-720, Artificial Intelligence Laboratory, Massachusetts Institute of Technology, 1983.

[3] Paul B. Chou and Christopher M. Brown. Multimodal reconstruction and segmentation with Markov random fields and HCF optimization. In *Proceedings Image Understanding Workshop*, pages 214-221, Cambridge, MA, February 1988. Morgan Kaufmann, San Mateo, CA.

[4] T. Poggio E. Gamble, D. Geiger and D. Weinshal. Integration of vision modules and labeling of surface discontinuities. *Invited paper submitted to IEEE Trans. Sustems, Man & Cybernetics*, December 1989.

[5] Edward B. Gamble and Tomaso Poggio. Visual integration and detection of discontinuities: The key role of intensity edges. A.I. Memo No. 970, Artificial Intelligence Laboratory, Massachusetts Institute of Technology, October 1987.

[6] Davi Geiger and Federico Girosi. Mean field theory for surface reconstruction and visual integration. A.I. Memo No. 1114, Artificial Intelligence Laboratory, Massachusetts Institute of Technology, April 1989.

[7] Davi Geiger and Tomaso Poggio. An optimal scale for edge detection. In *Proceedings IJCAI*, August 1987.

[8] Stuart Geman and Don Geman. Stochastic relaxation, Gibbs distributions, and the Bayesian restoration of images. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, PAMI-6:721-741, 1984.

[9] A. Hurlbert and T. Poggio. A network for image segmentation using color. *To be published in the Proceedings of the Denver Conference on Neural Networks*, November 1988.

[10] Christof Koch, Jose Marroquin, and Alan Yuille. Analog 'neuronal' networks in early vision. A.I. Memo No. 751, Artificial Intelligence Laboratory, Massachusetts Institute of Technology, 1985.

[11] Jose L. Marroquin. Deterministic Bayesian estimation of Markovian random fields with applications to computational vision. In *Proceedings of the International Conference on Computer Vision*, London, England, June 1987. IEEE, Washington, DC.

[12] Jose L. Marroquin, Sanjoy Mitter, and Tomaso Poggio. Probabilistic solution of ill-posed problems in computational vision. In L. Baumann, editor, *Proceedings Image Understanding Workshop*, pages 293-309, McLean, VA, August 1985. Scientific Applications International Corporation.

[13] Jr. S. Kirkpatrick, C.D. Gelatt and M.P. Vecchi. Optimization by simulated annealing. *Science*, 220:219-227, 1983.

[14] A. N. Tikhonov and V. Y. Arsenin. *Solutions of Ill-posed Problems*. W.H.Winston, Washington, D.C., 1977.

[15] G. Wahba. Ill posed problems. Technical Report 595, University of Wisconsin, 1980.

RAMBO—VISION AND PLANNING ON THE CONNECTION MACHINE

Larry S. Davis
Daniel DeMenthon
Thor Bestul
David Harwood
H.V. Srinivasan
Sotirios Ziavras

Computer Vision Laboratory
Center for Automation Research
University of Maryland
College Park, Maryland 20742-3411

1. INTRODUCTION

In this paper we describe research being performed on a system named RAMBO (**R**obot **A**cting on **M**oving **BO**dies). RAMBO resides principally in the "mind" of a Connection Machine, and drives a monocular camera and laser pointer (attached to a robot arm) through space. A second robot carries a target (perhaps along a "virtual" surface to mimic the motion of a vehicle along the ground) attached to whose surface are sensors (light-sensitive diodes) with focusing optics. RAMBO's general task is to illuminate a set or sequence of these sensors for specific durations of time, possibly subject to overall temporal constraints. Figure 1 is a diagrammatic representation of the experimental set-up.

The vision-based control loop for RAMBO is also shown in Figure 1. We briefly describe the functions of the different modules of this system, from data collection to robot motion control.

1. The digitizer of the video camera mounted on the robot arm can obtain video frames when new visual information is needed.
2. A low level Connection Machine vision module extracts the locations of the projections of model features (i.e., polyhedral vertices) from the image. Once a model for the target's motion has been established, the predicted locations of visible target features are established using fast table lookup procedures implemented in the Connection Machine.
3. An intermediate level vision module establishes the instantaneous pose (location and orientation) of the target in the camera coordinate system.
4. The target motion predictor fits a target trajectory in location/orientation space to the most recent history of instantaneous pose estimates. The trajectories of so-called ***goal points*** around the target (called ***goal trajectories***) are also determined. A goal point is a location fixed in the frame of reference of the target that one of the robot joints has to follow in order to accomplish one of the basic illumination subtasks of the total task. The determination of these trajectories should ideally take into account the subsequent visibility of a sufficient number of target features to verify or modify RAMBO's model of the target's motion, and safety criteria that would allow RAMBO to move away from the target to a safe viewing location in the event that the target's motion changes in an unanticipated way.
5. From the predicted goal point trajectories, the robot motion planner calculates the robot motions necessary for following the goal trajectories, and the resulting camera trajectories. If the subtasks were not ordered in the original task specification, then the motic:ı planner orders them (using either optimal or heuristic methods). The camera trajectories are used for transforming subsequent target pose estimates from the camera coordinate system to an absolute coordinate system. In our current implementation, the Connection Machine is used to plan a smooth motion from one goal trajectory to a subsequent goal trajectory (from which the next subtask can be performed)

The RAMBO project thus provides us with a context for studying several basic classes of problems in vision and visual navigation. These problems include the development of parallel Connection Machine algorithms for efficient image processing and analysis, visual tracking, and visual planning.

In order for RAMBO to complete even its most basic navigation task, it must be capable of visually tracking its target through space. Feasible tracking algorithms depend on many factors including sensor field of view, processing time per frame, relative motion of the target and the sensor, accuracy of sensor control, etc. We can

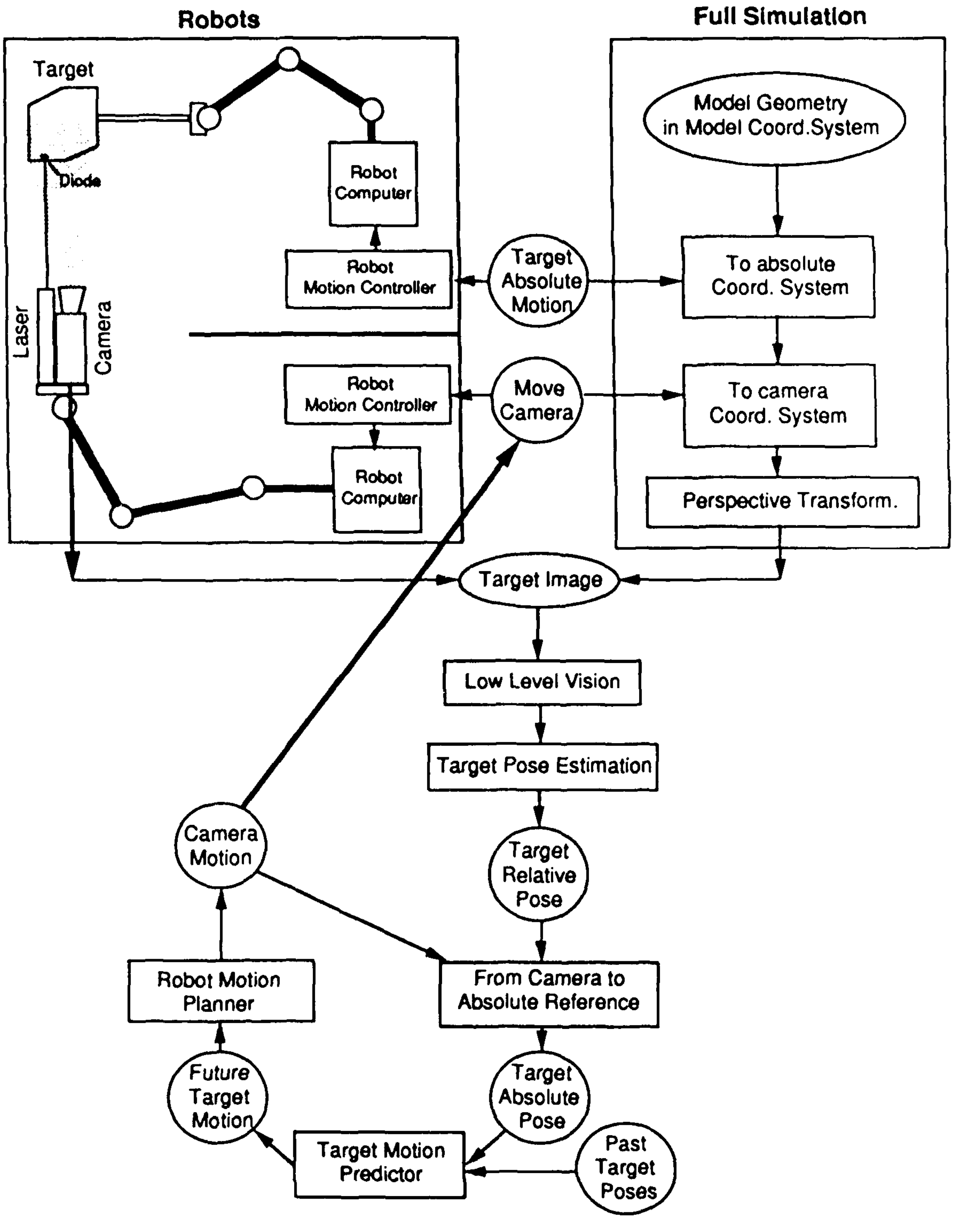


Figure 1. Vision-based control loop for a robot acting on a moving body

identify two basic approaches to visual tracking that are relevant to RAMBO:

a) Two dimensional tracking algorithms, in which the target can be kept in the field of view by determining its image motion, and computing a suitable sensor motion that "nulls" or minimizes the image motion (see Tsakiris and Aloimonos [Aloimonos and Tsakiris, 1988]).

b) Three dimensional tracking, in which a complete three dimensional model of the target's relative rigid body motion is established, and a sensor motion is determined that would cause certain components of the relative motion to be zeroed (for example, when RAMPO is firing its laser at the target, we might want all components of the relative rigid body motion between RAMBO and the target to be zero).

While approach (a) is simpler, it is not always applicable or sufficient. Its applicability depends on the factors listed above (i.e., sensor field of view, frame processing rate, etc.), although little research has been conducted that reveals the conditions under which two dimensional methods can be used for tracking. For example, given a frame processing rate, a model for the accuracy of image motion estimation and a model for the control accuracy of the sensor, one would like to know the minimum field of view that would guarantee (with some probability) that the target could be tracked.

However, even if it is possible to keep the target in the field of view using two dimensional methods, it might not be sufficient for the overall planning process. For example, RAMBO should compute its trajectory through space based not only its particular sequence of illumination tasks, but also based on its ability to retreat if the target changes its motion in a way that might potentially cause it to collide with RAMBO. This type of analysis can be more easily accomplished using direct three dimensional models.

One standard approach for three dimensional tracking is to compute a sequence of instantaneous target pose estimates, and then to fit a motion model to this sequence of pose estimates. The pose estimation problem arises in other applications as well (e.g., object recognition), so is of general interest. In Section 2 we describe a Connection Machine pose estimation algorithm. This algorithm, based on results originally presented by Linnainmaa, Harwood and Davis [Linnainmaa et al., 1988], involves generating target pose hypotheses from matches of triples of image features to triples of target surface features, and clustering appropriate low dimensional projections of the complete six parameter pose estimates. The algorithm is very fast, generating a pose estimate in generally less than one second of Connection Machine time.

The overall computational architecture of RAMBO is quite similar to the one we previously developed for road and road network navigation [Sharma and Davis, 1988; Waxman et al., 1987]—with separate computational modules for image processing, geometric reasoning, sensor control, motion planning and plan supervision (see Figure 1). RAMBO's task set, however, leads to a much richer set of problems in visual planning. In Section 3 we describe how RAMBO determines an appropriate subtask ordering if the initial task definition does not specify a fixed subtask ordering, and also explain how the Connection Machine can be used to establish smooth motions between consecutive goal trajectories.

2. POSE ESTIMATION

In the current implementation of RAMBO, the relative rigid body motion of the target is determined by fitting a polynomial model to a sequence of instantaneous pose estimates. In a "feedforward" mode of processing, a new pose estimate can be quickly computed based on predictions of the image projections of specific target surface features. However, in order to "bootstrap" this procedure, or to recover from gross errors due to either changes in the trajectory of the target or mistakes in image analysis, we have developed a Connection Machine pose estimation algorithm that is not based on any prior knowledge of target pose or motion.

This algorithm is based on work originally described in Linnainmaa et al. [1988]. In that paper we showed that if a triple of image features (i.e., perspective projections of polyhedra corners) could be matched to a triple of target surface features, then a simple quartic equation can be solved to determine a small number of six degree of freedom pose estimates (in fact, the equations almost always have only two solutions). Since it is difficult to determine which image features match to which target features in the absence of prior knowledge, the basic hypothesis generation procedure is embedded in a clustering algorithm that matches many combinations of three image features against combinations of three target features. Various heuristics can be employed to reduce the combinatorics of this matching process. The key to the success of the clustering process is the choice of an appropriate projection of the six dimensional pose space in which to perform the initial clustering. The projection used was a two dimensional projection corresponding to the visual direction to the target center under any hypothetical pose estimate. Within each bin of this two dimensional clustering space, pose estimates were grouped by visual size of the target.

A straightforward implementation of this algorithm on the Connection Machine would result in a very slow pose estimation process because of the intensive floating point arithmetic operations associated with solving the quartic equations determined by each image triple/target triple combination. Below, we describe how a few simplifying assumptions allow us to replace these arithmetic operations by a small number of table lookup operations. The result is a pose estimation algorithm that takes less than one second of Connection Machine time for images of our target.

2.1. ALGORITHM DETAILS

Our Connection Machine pose estimation algorithm combines three ideas:

1) Pose estimation by matching triples of image features to triples of target features [Linnainmaa et al., 1988].

2) Standard camera rotations [Kanatani, 1988].

3) Paraperspective approximation to perspective projection [Aloimonos and Swain, 1987].

This combination allows the extensive use of lookup tables in lieu of costly floating point arithmetic operations. Feature points detected in the image are grouped into triples (called *image triangles*). Each image triangle is defined by a distinguished vertex (called the *reference vertex*), the lengths of the two adjacent sides, and the angle between them (the *reference angle*). Image triangles can be computed for all triples of detected image features (three triangle per triple, since any point can be chosen as the reference vertex) or various heuristics can be employed to reduce the number of image triangles constructed (for example, our current implementation includes a simple test for vertex connectivity).

The determination of the target's pose is somewhat simplified if each image triangle is first transformed so that its reference vertex is at the image center and one of its edges is coincident with the image x-axis. This is equivalent to a rotation of the image plane (with the rotation parameters expressed as functions of the reference vertex's initial image position) to bring the reference vertex to the image center, followed by a camera roll to bring one edge into coincidence with the x-axis. In [Kanatani, 1988] Kanatani developed simple formulas for these rotations.

Consider a given image triangle/target triangle pair after the image triangle has been rotated into standard position—see Figure 2. The position of the target triangle in space is then determined if we can compute:

1) Angles θ_1 and θ_2 of the lines P_0P_1 and P_0P_2 with respect to the Z-axis.

2) Distance, R, from the center of projection, O, to the vertex P_0 on the Z-axis.

The perspective transformation which maps the target triangle onto the image triangle is approximated by a paraperspective projection [Aloimonos and Swain, 1987]. This amounts to a sequence of two transformations—a local orthographic projection of the target triangle onto a plane parallel to the image through P_0, and a perspective projection of the resulting image onto the image plane. This is depicted in Figure 3. From this Figure we can see that:

$$f/R = d_1/(D_1 \sin\theta_1) \tag{1}$$

and

$$f/R = d_2/(D_2 \sin\theta_2) \tag{2}$$

We then define the parameter s, the ratio of the lengths of the two sides of the image triangle; S, the ratio of the lengths of the two sides of the target triangle; and K, the ratio of s and S:

$$s = d_1/d_2;\ S = D_1/D_2;\ K = s/S$$

The parameter K is thus a known constant for any image triangle/target triangle pairing. Dividing (1) by (2) we find that K is the ratio of the sines of θ_1 and θ_2:

$$K = \sin\theta_1/\sin\theta_2 \tag{3}$$

The dot product of the two unit vectors $\mathbf{n}_1$ and $\mathbf{n}_2$ parallel to P_0P_1 and P_0P_2 is equal to $\cos\alpha$. These two unit vectors have components $(\sin\theta_1,\ 0,\ \cos\theta_1)$ and $(\sin\theta_2\cos\rho,\ \sin\theta_2\sin\phi,\ \cos\theta_2)$, where ϕ is the angle p_0p_2 makes with the image x-axis. The resulting dot product is therefore

$$\cos\alpha = \sin\theta_1\sin\theta_2\cos\phi + \cos\theta_1\cos\theta_2 \tag{4}$$

The angles θ_1 and θ_2 are unknown. The fact that the ratios of the sines of these angles must be equal to the known constant, K, allows us to eliminate one of the unknowns from (4). This yields a second degree equation

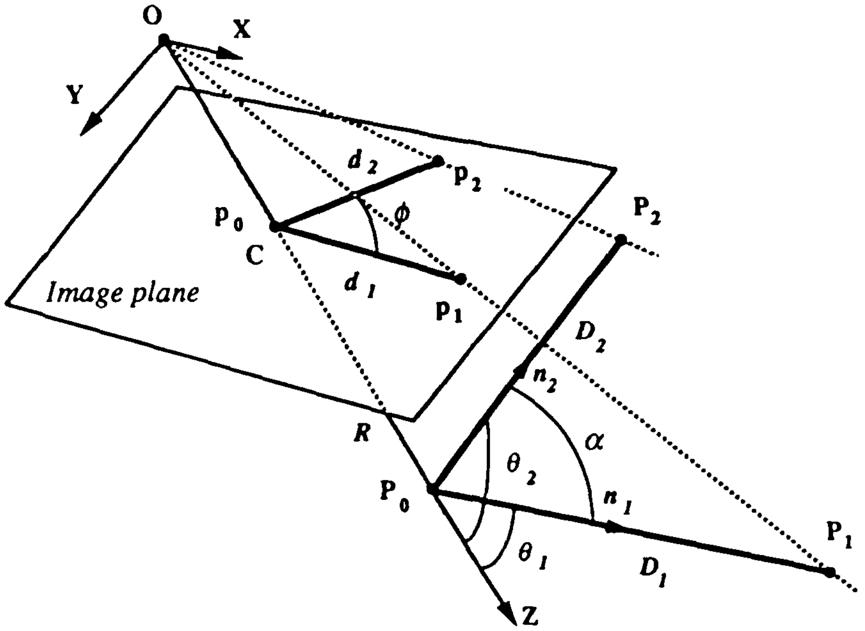


Figure 2. Image triangle with vertex at image center, and pose required for a given world triangle to match this image triangle

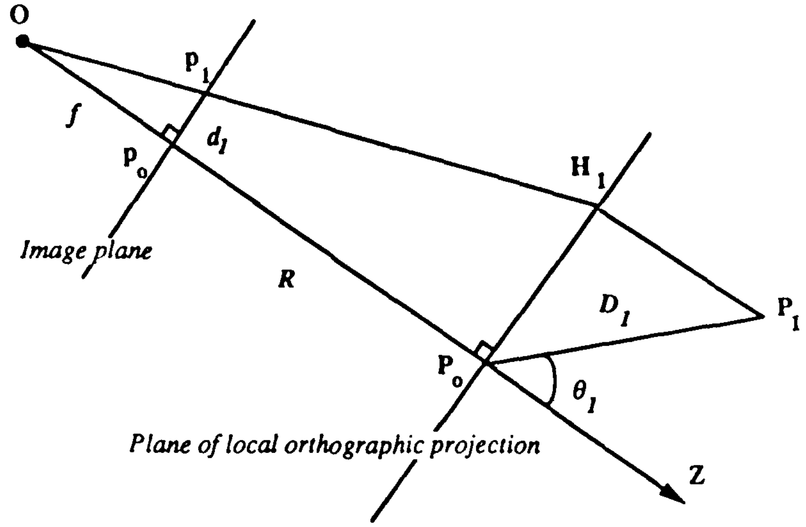


Figure 3. Perspective transformation of triangle side *PoP1* approximated by a paraperspective transformation giving image triangle side *pop1*.

in $\sin^2\theta_1$. If we let $X_1 = \sin^2\theta_1$, then this equation can be written as

$$\sin^2\phi X_1^2 - (K^2 - 2K\cos\alpha\cos\phi + 1)X_1 + K^2\sin^2\alpha = 0 \tag{5}$$

This equation always has two positive solutions, but only the smaller of these solutions has magnitude less than 1 and can thus be equated with the sine of an angle. Thus, the solution of this quadratic always yields a single solution for $\sin\theta_1$. This single sine solution results in two solutions for θ_1 differing by π. They correspond to mirror image directions of P_0P_1 with respect to the plane parallel to the image plane through P_0. The corresponding value for $\sin\theta_2$ obtained from (3) also yields two solutions for θ_2 separated by π. Note that we cannot combine the two solutions for θ_1 with the two solutions for θ_2 arbitrarily. Two of these combinations yield a positive number for $\cos\theta_1\cos\theta_2$; the other two combinations result in a negative product. One of these pairs is unacceptable and was introduced when (4) was squared. Only the two combinations consistent with (4) are allowable solutions. Finally, for each allowable pair (θ_1, θ_2) we compute the distance, R, of the vertex P_0 from the center of projection. Once θ_1, θ_2 and R are known, we can determine the three dimensional location of any other target point. Specifically, we compute the location of the centroid of the target, and then compute its projection onto the image plane. We apply the inverse transformations of the roll and standard rotation (that brought the reference vertex to the image center to the target centroid projection) to compute the direction of sight of the target under this particular pose hypothesis. A two dimensional array of possible centroid projections is maintained, and at each location in the array we both count the number of image triangle/target triangle pairs that yielded pose estimates resulting in that projected target centroid location, and we maintain a list of those pairs along with the distances computed to the target centroid. This information is used by a subsequent clustering algorithm to identify large subsets of image triangle/target triangle pairs yielding sufficiently similar pose estimates. The details of that clustering algorithm, sketched below, can be found in DeMenthon, Ziavras and Davis [DeMenthon et al., 1989].

2.2. CONNECTION MACHINE IMPLEMENTATION

The Connection Machine is used in three ways to implement the pose estimation algorithm:

1) as a lookup table engine;

2) as a combinatorial machine, considering all combinations of image triangles and target triangles;

3) as an image processor for calculating convolutions and finding peaks (clustering) in the Hough transform space.

2.2.1. Lookup Table Engine

There are several different table lookup operations performed in the course of pose estimation. First, the parameters of the standard rotation are stored in a two dimensional lookup table indexed by image position. The parameters of the inverse rotations are also stored in this two dimensional lookup table.

A second set of lookup tables is maintained, one for each possible target triangle. These are also two dimensional lookup tables, indexed by α, the reference angle of an image triangle and K, the ratio of s and S. Each such table contains the three dimensional location of the target centroid as output (there is no need to explicitly store or compute the intermediate variables corresponding to the orientation and location of the target triangle in the image coordinate system).

All of these tables can be computed beforehand and loaded into the Connection Machine.

2.2.2. Combinatorial Machine

Here we describe how the image triangle/target triangle pairs are distributed in the Connection Machine. We regard the Connection Machine as a two dimensional array. Each target triangle is assigned to one row of this array. The coordinates of P_0 and the value of S (for indexing into the corresponding target triangle table) are stored with each copy of the target triangle. Image triangles are assigned to columns of the array. The information initially associated with each image triangle includes the image plane coordinates of its vertices and the parameters of both the standard rotation and its inverse. The target triangle data is then scanned across the rows and the image triangle data is scanned up the columns to create the combinatorial pairing of image triangles/target triangles.

2.2.3. Image Processor

The analysis of the pose estimate voting patterns of the image/target triangle pairs involves operations common to basic image processing. The two dimensional clustering array of projected target centroids is represented in the Connection Machine by assigning one processor per location of this two dimensional array. After all votes are cast by the image/target triangle pairs, the counts in this array are locally smoothed, and the smoothed array is then thresholded. The above threshold processors are then numbered according to their vote strength, and the subset of image/target triangle pairs that contributed to the above threshold counts are selected for further processing.

A second clustering step is then applied, in parallel, to the triangle pairs corresponding to each above threshold centroid projection. Each centroid projection is assigned to a row in a two dimensional matrix, and the triangle pairs that contributed to that centroid projection are then loaded into the columns of that row and bucket sorted by Z coordinate of the centroid. Each row is smoothed independently, and the highest cluster is finally selected as the correct cluster. The triangle pairs that contributed to that cluster are then selected, and a final least squares estimate of the target's pose is computed based on the actual correspondence of image features to target features.

3. TASK AND TRAJECTORY PLANNING

In order to perform task and trajectory planning, RAMBO currently makes the simplifying assumption that a complex goal can be decomposed into a sequence of simple subgoals, and that each subgoal can be performed with one joint of the robot in a fixed position with respect to the target. This joint has to "tag along" with the target and so we call it the *tagging joint* of the robot. The fixed position with respect to the target that the tagging point must follow to complete a subgoal is called the *goal point*. All goal points required for each complex action on a target can be predetermined and stored in a database of actions specific to each target. Each goal point is defined by its pose in the target coordinate system.

As RAMBO proceeds from one subgoal to another, it will generally have to change the trajectory along which it moves, so that at some time t_0 we would want RAMBO to launch from its current goal trajectory and to land at some subsequent time, $t_0 + T$, on a new goal trajectory. The duration T is referred to as the *reaching duration*. Once t_0 and T are chosen, then a *reaching trajectory* that takes RAMBO from its original goal trajectory to the new goal trajectory can be determined by using, for example, a parametric cubic spline that ensures continuity and smoothness at both takeoff from the original trajectory and landing on the new goal trajectory (see Figure 4).

A subproblem, then, in controlling RAMBO's motion from one trajectory to another is the choice of T, the transit time. It should be chosen so that the resulting linear and angular velocities and accelerations are within the limits of RAMBO's motions. Additionally, the resulting reaching trajectory should be safe in the sense that it should not cross the path of the target and should not require RAMBO to assume impossible configurations. The Connection Machine can be used to examine a range of reaching durations in parallel, finally choosing the smallest reaching duration resulting in a realizable reaching trajectory.

We set up a 2D array of processing cells with time as the vertical dimension, and values of T as the horizontal dimension. Each column of the array contains a copy of the predicted goal trajectory, with the first row containing the position of the goal at the present time in location/direction space, the next row containing the position at a time increment beyond that, etc. Every column also contains a copy of the trajectory of the tagging joint from the original trajectory, sampled with the same time increments as the goal trajectory. The difference between columns is that they use different durations, T, of the reaching trajectory, increasing from one column to the next.

Each cell in the array computes a point of the reaching trajectory for the time t_0 corresponding to its row and for duration T corresponding to its column. It then computes estimates of appropriate derivatives of its reaching trajectory by communicating with its neighbors in the column. The maxima of the derivatives are computed for each column and the column that has the smallest T for which the maximum derivatives are within bounds is chosen to determine the reaching trajectory. The near term future motion of the robot should be controlled based on this selected trajectory.

A rule-based system is used for completing sets of tasks. It is based on a heuristic strategy for dealing with illegal trajectories (ones involving collisions or impossible robot positions or motion derivatives) and incorporates either a greedy strategy for choosing tasks or a fixed task ordering. Both versions are described below. Essentially, the following ordered set of rules is iteratively applied until the set of tasks is completed. The rules were

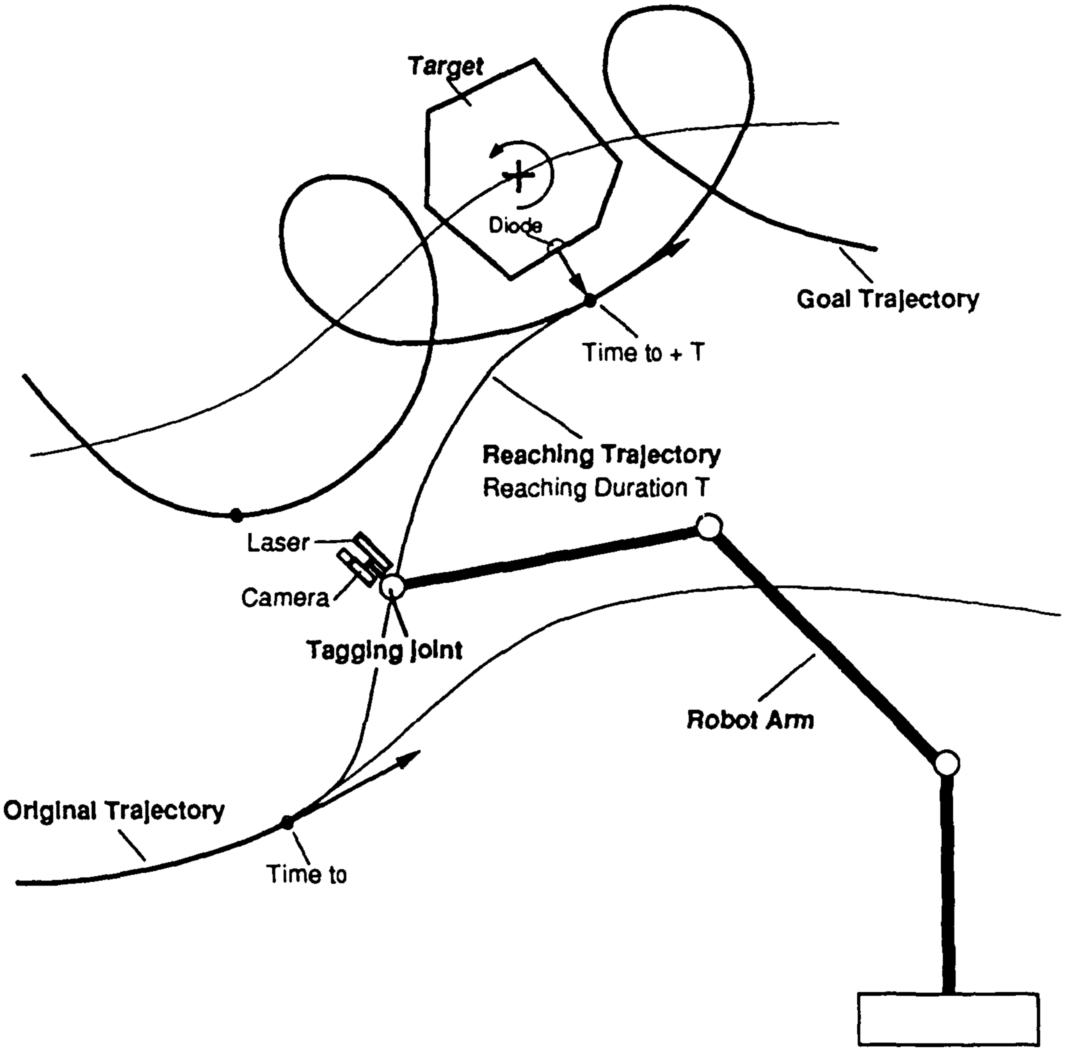


Figure 4. A tagging joint takes a reaching trajectory during time *T* to reach a goal trajectory required to grip a handle on the target

chosen to be simple, but fairly robust.

Rule 1: If currently on the goal trajectory of an uncompleted task, remain on it for the specified task duration, then restart rule set.

Rule 2 (Greedy): Find reaching trajectories to all remaining goal trajectories. Choose quickest legal reaching trajectory to a goal trajectory and follow it until reaching the corresponding goal trajectory, then restart rule set.

Rule 2 (Fixed order): Otherwise, find reaching trajectory to the goal trajectory of the next task in the desired sequence. Follow this reaching trajectory until reaching the corresponding goal trajectory, then restart the rule set.

Rule 3: If there is no legal reaching trajectory to a goal trajectory, find a reaching trajectory to the "approach" trajectory (this is a pre-defined trajectory relative to the target motion from which some goal trajectory is likely to be reachable in the future), begin to follow this reaching trajectory and restart the rule set.

Rule 4: If the reaching trajectory to the approach trajectory is not legal, maintain present position relative to the target and restart the rule set.

If a revision of the target motion model parameters occurs during the application of any of the rules, the rule set is restarted. This is because a revision of the target motion will sometimes cause a reaching trajectory or section of goal trajectory previously thought to be legal to be disallowed. Most of the time, however, since the revisions of the motion model parameters will not be large, there will be no drastic change in the robot motion due to them.

4. CONCLUSIONS

We have described progress to date on constructing a set of Connection Machine vision and planning algorithms that should allow RAMBO to plan, monitor and execute a complex navigation task. These algorithms are currently being integrated so that they may be tested on some simple initial navigation tasks. Additionally, we are developing fast Connection Machine two-dimensional target tracking algorithms, which could be interleaved with the more computationally demanding three dimensional tracking algorithms, and studying the applicability of logic programming methods for specifying, synthesizing, monitoring and controlling RAMBO's actions.

REFERENCES

[1] J. Aloimonos and M. Swain, Paraperspective projection: between orthography and perspective, University of Maryland Center for Automation Research Technical Report 320, May 1987.

[2] J. Aloimonos and D. Tsakiris, On the mathematics of visual tracking, University of Maryland Center for Automation Research Technical Report 390, September 1988.

[3] D. DeMenthon, S. Ziavras and L. Davis, Connection Machine pose estimation, University of Maryland Center for Automation Research Technical Report, in preparation.

[4] K. Kanatani, Constraints on length and angle, *Computer Vision, Graphics and Image Processing*, **41**, 1988, 28-42.

[5] S. Linnainmaa, D. Harwood and L. Davis, Pose determination of a three dimensional object using triangle pairs, *IEEE Transactions on Pattern Analysis and Machine Intelligence*, **10**, 1988, 634-647.

[6] C. Sharma and L. Davis, Road boundary detection in range imagery for an autonomous robot, IEEE **4**, 1988, 515-523.

[7] A. Waxman, L. Davis, et al., A visual navigation system for autonomous land vehicles, *IEEE Trans. On Robotics and Automation* **3**, 1987, 124-141.

MAC-II BASED IU ENVIRONMENT DEVELOPMENT

Philip Kahn, David Disabatino, Daryl T. Lawton
Advanced Decision Systems
1500 Plymouth Street
Mountain View, California 94043-1230

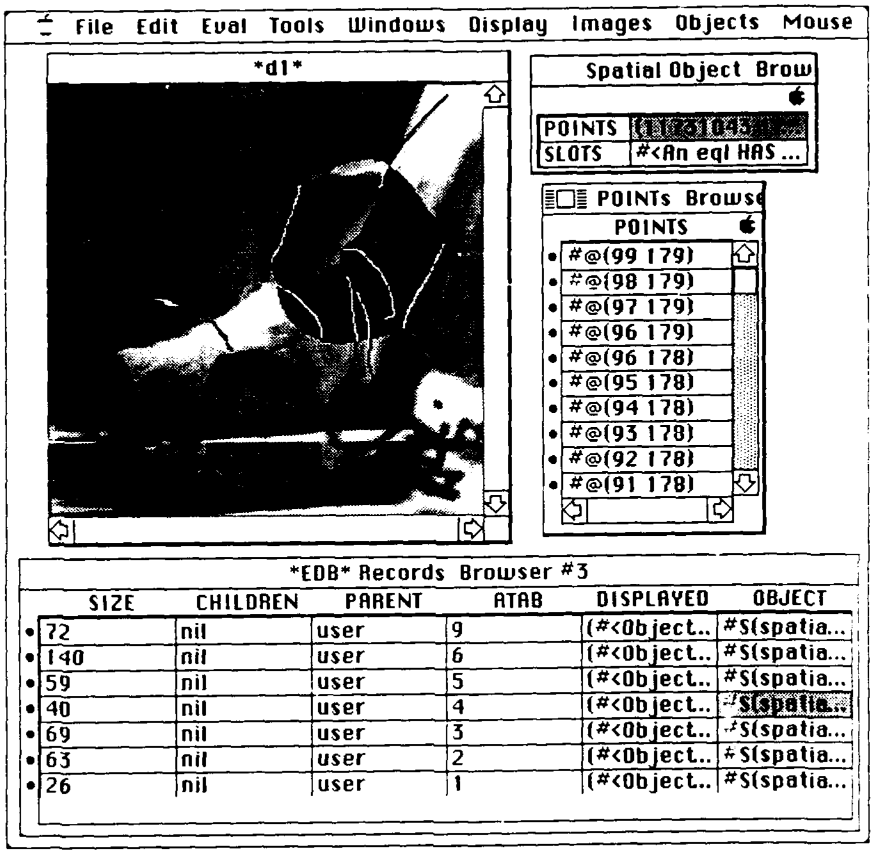
1. INTRODUCTION

Image understanding software environments are important tools for supporting research, development and technical transfer. They are used to support shared development across multiple researchers and projects and to make possible the rapid prototyping of applications and experiments. The major objective of the work described here is to construct a low cost image understanding environment, along the lines of the general architecture laid out in [Lawton and McConnell - 88, McConnell et.al. - 87], that would run on low-cost personal computer workstations. A basic goal is to stress machine independence and also to align the development effort with ongoing and extensive commercial activity in coprocessors, video technology, programming environments, and available software. This will enable applications to extend the core environment using commercially or community-supported tools whenever possible and will enhance modularity.

We have initially targeted a Mac II as a platform for this development due to it's availability and the large number of commercial products that exist for it. The Mac is an open architecture which is readily extended by a large, and rapidly growing, number of coprocessors, storage and video boards, cameras and digitizers. It is fairly simple to construct, with off-the shelf hardware products described below, the basic components of an image processing laboratory for image access, storage, display, and processing. This is also true for software products. There is an enormous number of commercially developed and supported software packages that are available. For example, a voice activated, hypermedia-based interface can be constructed using Silicon Beach's Supercard and Articulate System's Voice Navigator for around $1500. The Macintosh also supports an extensive, well-tested, and optimized user interface and construction facilities built around the ROM-based set of Toolbox routines [Apple Computer - 88]. This includes such things as windows, browsers, controls, overlays, and dialogs.

The basic components and constructs used in our environments have been described in [Lawton and McConnell - 88, McConnell et.al. - 87, Edelson, et.al. - 88, Riley et.al. - 87]. In general, we found that developing corresponding components of the user interface on the Mac was fairly simple. For example, Figure 1 shows the interactive application of a spatial mask to a set of edges registered with an image. The selected edges are then displayed in a Browse Table for further inspection Figure 2 shows the association of a Pixel-Mapping-Function between linked windows, where one window is zooming onto the selected portion of the other. We found that several of the commercially available products could be integrated with image handling. The use of generic CommonLISP made possible the use of programming constructs developed for CommonLISP-based IU environments.

On the negative side, we found that integrating other commercial application software such as CAD/CAM, databases, and expert system building tools was still rather difficult and awkward. In part, this is due to the lack of a conventional operating system which supports such things as virtual memory, multi-tasking,



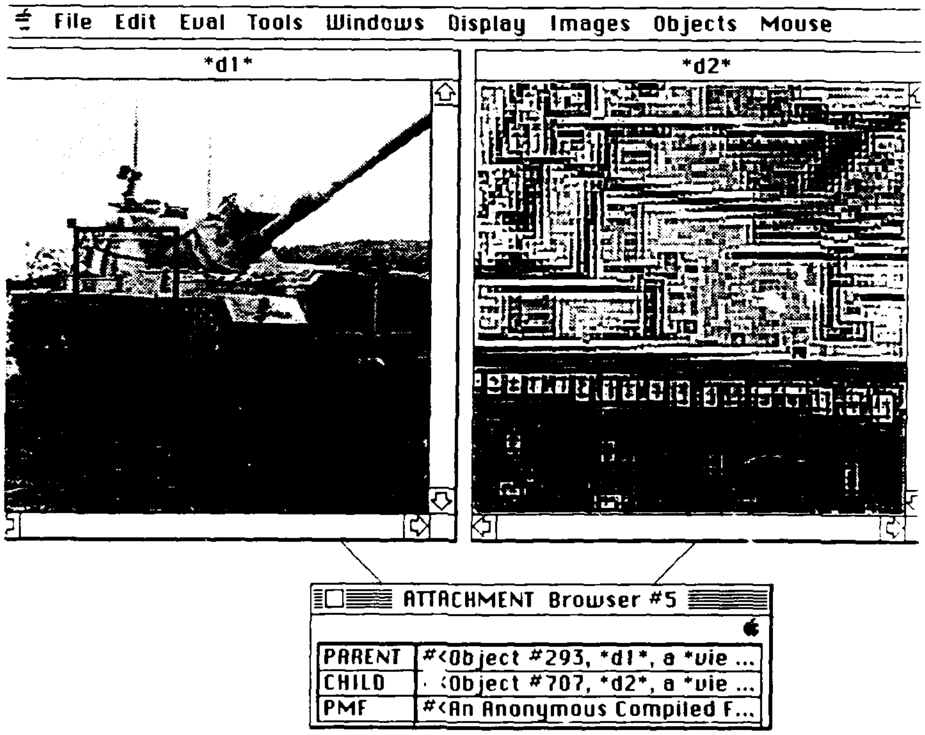
POINTS	[illegible]
SLOTS	#<An eql HAS ...

POINTS
#@(99 179)
#@(98 179)
#@(97 179)
#@(96 179)
#@(96 178)
#@(95 178)
#@(94 178)
#@(93 178)
#@(92 178)
#@(91 178)

EDB Records Browser #3

SIZE	CHILDREN	PARENT	ATAB	DISPLAYED	OBJECT
72	nil	user	9	(#<Object..	#S(spatia...
140	nil	user	6	(#<Object..	#S(spatia...
59	nil	user	5	(#<Object..	#S(spatia...
40	nil	user	4	(#<Object..	#S(spatia...
69	nil	user	3	(#<Object..	#S(spatia...
63	nil	user	2	(#<Object..	#S(spatia...
26	nil	user	1	(#<Object..	#S(spatia...

Figure 1: Selection and Browsing of Curve Objects



PARENT	#<Object #293, *d1*, a *vie ...
CHILD	· <Object #707, *d2*, a *vie ...
PMF	#<An Anonymous Compiled F...

Figure 2: Attachment Relation with Pixel Mapping Function

interprocess communication, and the lack of language interfaces for much of the commercial Mac software. Another limitation that has made us question the appropriateness of the Mac is the Allegro Coral Common-LISP environment itself. While providing great flexibility and power as a development environment, it needs to be greatly optimized as a run-time system before we can consider it for developing computationally intensive IU environment applications. The main hardware limitations we experienced stem from an overworked CPU. The central processor must not only perform all the computation and symbolic processing required for image understanding, but must also drive all the graphic display operations that are so fundamental to this type of interactive image and graphical object oriented application and orchestrate the access of all devices to the main memory. The CPU handles all file writes and memory management tasks byte by byte. This problem can be solved in several ways, and rumors from Apple indicate that they are actively developing hardware that addresses all these solutions (e.g., using an AMD 29000 RISC graphics coprocessor increasing bus bandwidth from 10mHz to 20mHz, and having direct memory access channels).

The limitations we have discovered are, in all likelihood, soon to be addressed by Apple so they can compete with low-cost scientific workstations from other vendors such as Sun, DEC, Apollo, and Next. With the increasing sophistication of user-interface tools associated with these machines, increased power, and a large third-party supply of hardware and software, they all provide very attractive platforms for IU environments.

2. SYSTEM CONFIGURATION

There are three primary components in an IU environment: IU constructs and structures, user interface, and the underlying hardware platform and system-level support.

IU constructs and structures provide the means for manipulating imagery and extracted features, extracting and describing the content of imagery at varying levels of abstraction, input/output, etc. As in any language, the expressiveness and power of IU structures and constructs largely determines the naturalness, expressiveness, and efficiency with which vision and image analysis processing may occur.

An IU environment user interface provides the means by which a user of the system may flexibly and comfortably interact with data, understand and debug algorithms, and modify the state of the processing and system. We have found that a highly interactive user interface is most productive and is usually required for the user to efficiently develop, understand, and describe processing algorithms and results. Inflexible, highly textual, or low-bandwidth user interfaces significantly and directly decrease the power and flexibility of the IU environment user. Typical elements in a flexible IU environment u[illegible]rface include rapid graphical display (e.g., pan, zoom, drawing), color graphical overlay (e.g., sho[illegible]es in an overlay plane), powerful window and menu control mechanisms, flexible mouse-driven processing (e.g., when the mouse is clicked over an edge, a user-selected function is executed with that edge as an argument), data and environment browsing facilities (e.g., inspecting the values in an image or in a collection of extracted image objects), user-interface building tools (e.g., for building domain-dependent menus, specifying mouse-driven processing, complex text and numeric input via window dialogs with the user, etc.), and other tools to ease the burden and increase the expressiveness of the user. Other sensory modalities (e.g., computer generated cues or speech, voice keyword recognition, touch screens) can also enhance the power and efficiency of the user interface.

A flexible and fairly powerful IU environment user interface was prototyped in Allegro Coral Common-LISP (CCL) on the MacII. The object-oriented tools provided by CCL and the native Mac graphical routines and hardware greatly simplified the design and coding of this prototype IU environment interface. Though the interface is fairly fast, our exclusive use of CommonLISP and the use of some high-level interface mechanisms provided by CCL somewhat decreased the performance which can otherwise be achieved by using C and more directly accessing the underlying Mac graphics primitives.

The Mac-based IU environment development hardware and system level support has several goals. First, to provide a powerful and extensible front end for the IU software environment. Second, the base level of equipment should be inexpensive (about $10K). Third, the system should easily scale up into more advanced applications by incorporating commercially available add-on modules.

The Mac IIx and Mac IIcx are full 32-bit, 68030-based machines using a 15.67mHz clock, a 68882 floating point coprocessor, SCSI and RS-422 ports, and a flexible NuBus expansion bus architecture (10mHz). The Mac IIcx has three NuBus expansion slots, and the Mac IIx provides six NuBus expansion slots. The NuBus provides an open architecture which has led to widespread commercial availability of powerful and inexpensive extended hardware capabilities which include coprocessors, memory, storage, digitizers, extended ports, robotic control, measurement, voice recognition, and a rapidly growing assortment of other equipment. Because of the large commercial Mac market, these products are relatively inexpensive, powerful, simple to install, and well supported.

The next sections concentrate on the description of a low-cost and modular Mac-based hardware platform for IU environments. The processing and interface requirements of an IU environment on the development and design of this underlying system are discussed.

2.1 BASE-LEVEL SYSTEM

The base level system is intended for the entry level user who wishes to have basic vision and image processing capabilities at very low cost (about $10K). Basic capabilities include the ability to read in imagery, perform advanced image and vision processing, display and store imagery and results, and share data and programs with a networked computing community and file system. Main components of the base level system include the IU software environment and interface, basic computing equipment, basic secondary storage facilities, display capabilities, networking, and primary system level software and utilities.

The base level system is designed so that it can be simply scaled up by adding off-the-shelf commercial hardware and software modules. For example, enhancements in processing power, storage, graphics display, voice keyword interface, and extended capabilities for hard copy production, image acquisition, camera control, robotics, and mobile capabilities can be achieved by adding commercially available modules which can be added to the core system.

A base level system should be able to perform simple vision processing tasks in a reasonable amount of time, provide for the basic user interface, provide for reasonable secondary storage and archival, and network and share files with the larger computing community. The base level hardware to support this includes a Mac IIx or Mac IIcx with 8MB memory, secondary storage comprised of an 1.4MB floppy drive and at least an 80MB hard drive, an Ethernet card, and a basic color monitor and video card (640 x 480 x 8-bits). The retail price for this equipment is about $10K.

Secondary storage for the base level system includes an 1.4MB internal Apple floppy drive and at least an 80MB hard disk drive. The floppies can be used to archive or transport small amounts of data, and the

Ethernet provides high speed transport of data to other machines or file servers on the network. Large scale archival may be achieved using the Ethernet to transfer data to other machines on the network which have high volume archive media (such as tape drives). If all system software occupies about 30MB (including source), this leaves at least 50MB for user files and data. This provides a workable amount of storage for most vision and image processing tasks.

An inexpensive graphical display for the base levei system can be provided by a 13" Apple coior monitor with 8-bit video card (640x480). An Apple extended keyboard and mouse facilitate user input. Alternative displays, cards, keyboards, and mice are available at comparable cost, and the particular choice is subject to individual user preference. Greater display resolution is simply provided by using alternative third-party monitors and video cards. Graphics display is standardized by the Mac II operating system, and software written according to this standard is independent of changes in display hardware. Thus, changing monitor and display resolution should not require display and interface code modification. Additionally, the number of monitors is only limited by the number of open NuBus expansion slots, and advanced multimonitor features are supported by the operating system (e.g., windows may be dragged from one monitor to another).

2.2 MODULAR EXTENSIONS TO THE BASE LEVEL SYSTEM

2.2.1 Display:

Fidelity and flexibility of graphic display is particularly important for advanced vision research and image analysis applications. Display feedback is an important tool for debugging and evaluating algorithms and code. Though the Mac II allows as many monitors to be used as there are available NuBus slots (up to six), we have found that a two monitor system provides good cost performance: a moderate to high resolution color console monitor and a high resolution full color monitor. The console monitor requires only moderate resolution, it provides the primary textual feedback to the user, and it allows moderate resolution display of imagery. The display monitor is primarily used for very high fidelity display of imagery and image objects (e.g., color or black and white images, edges, surface plots, target identification, etc.). A display monitor should have high spatial and color resolution in order to avoid introducing visual anomalies in displayed data.

The base level system provides the moderate to high resolution color monitor (640 x 480 x 8-bits). Commercially available high resolution monitors (e.g., 1024 x 768) are available for about $4K retail and the color video driver cards run from about $1500 (8-bit) to $4K (24-bit) retail.

2.2.2 Storage and Archival:

The large amount of data encountered in vision research and image analysis justifies the use of large disk drive units which can store the large quantities of intermediate processing results, imagery, and programs. Secondary drives are also related to archive capabilities, since removable drive media can provide good archival facilities. Removable archival media is required when imagery and processing results are to be stored for future use or shared among distant users, and to provide backup facilities for data recovery. A fast and easy archival media also reduces the size and cost of required magnetic drives.

There are three main types of secondary/archival devices: tape, removable magnetic disk, and removable optical storage. Tape archival provides for high volume inexpensive backup, but the sequential access excludes its use as a viable secondary storage. Cost for these tape backup units varies from about $1K for a 40MB cartridge to several thousand for a reel-to-reel or high density 8mm units. Removable magnetic disk media has the advantages of random access and low access time (e.g., 20ms), which allows its use as secondary storage, though these disks tend to be small and have a very high relative cost per MB. Optical media has provided an attractive alternative. Write-once-read-many (WORM) optical drives are fairly inexpensive, random access, and possess access times approaching slow hard disks. Yet, the write-once is generally a limitation that is not well-suited for the vision and image analysis domains. A very good solution in terms of cost price performance is afforded by the new magneto optical removable media drives which have reasonable access times (e.g., 50ms), good transfer rates (e.g., 1.2MB/sec), large capacity (e.g., 600MB), multiple read-write as offered by conventional magnetic drives, relatively inexpensive media (e.g., $268 per 600MB), and relatively low drive unit cost (e.g., $4000). These drives are have been newly introduced by Ricoh, Sony, and Canon.

Of course, the particular choice of secondary/archive method depends upon required access times, quantities of data, cost limitations, etc. of the particular problem domain. The main point to be made is that a wide range of viable choices are available as modular additions to the core system.

In addition to the secondary storage device itself, NuBus direct-memory access (DMA) boards are newcomers to the Mac market which greatly improve disk and overall system performance. These boards essentially contain a very fast SCSI chip set, fairly large cache memory, and a NuBus controller. Hard drive I/O thus ties up less CPU time, and the fast SCSI interface provides higher data transfer rates. Because the NuBus controller in some of these boards supports burst mode (which the Mac II does not), other NuBus boards which support burst mode can greatly benefit from such a board. These DMA boards vary from $500 to several thousand depending upon the amount of memory placed on the board.

2.2.3 Image Acquisition:

In order to experiment and algorithmically manipulate imagery from a program, images must first be converted to a digital form. The acquisition of image data is often a difficult, cumbersome, and time-consuming task. A frequent problem limiting many vision researchers is their inability to flexibly acquire real image data in order to test and demonstrate algorithms and techniques. This problem is especially difficult for motion researchers since a large amount of imagery must be collected from a sequence of video frames. To simplify these tasks, image acquisition system modules may be easily added to the base level system.

A full discussion of image acquisition hardware and technologies is beyond the scope of this paper, so we instead overview the range of possible image acquisition solutions. There are two main methods for acquiring imagery: video input or scanned from hardcopy (e.g., photographs).

Video input requires a video digitizer and generally an NTSC video source (e.g., as produced by most video cameras in the U.S.). Other video formats are possible (e.g., RS-170, HDTV). Good quality real-time monochrome digitizers for the Mac II retail for about $1200, and color digitizers have recently become widely available for less than $3K. Video sources may be obtained from videotape (e.g., 3/4", consumer VHS), videodisc players, live cameras, or video generators. Videotape as a playback media introduces anomalies and distortions into imagery, though a time-based corrector which retails from $2K-10K (depending on quality) may be used to partially offset some distortion and freeze frame problems. More expensive (i.e., $40K and up) videotape players have accurate single frame access capabilities, though the prices seem to be coming down somewhat. Videodisk recorders/players provide single write, random access, good-freeze frame capability, built-in RS-232 control to allow simple computer integration, good monochrome resolution (B/W resolution is about 450 lines, color resolution is about 350 lines), reasonable storage (e.g., 16,000 frames) and rela*ively low cost (e.g., $12K). For example, the Panasonic TQ-2026F unit offers these features. The best quality source of video data comes directly from a good quality CCD videocamera (monochrome CCD videocameras retail from several hundred dollars to about $3K for smaller size, higher resolution, and added features). The videocamera output may then be directly fed into the video digitizer. Without exception, this is the preferable method for video image acquisition when image quality is of concern.

Alternatively, images may be acquired from hardcopy image sources (e.g., slides, photos, books). We have not considered very high end laser scanners which produce the best quality digitization from hardcopy, since their cost can exceed $100K. Desktop scanners provide a lower cost, good quality (e.g., 400dpi) alternative. These scanners operate much as a photocopier, and they produce good spatial resolution and digitization. When simplicity and better spatial resolution is important, scanners are often preferable to video acquisition systems. An 8-bit B/W 300dpi desktop scanners retail for less than $3500, and 24-bit color digitizers retail for about $6K.

2.2.4 Coprocessing:

Overall computational power can be increased by orders of magnitude with the addition of commercially available coprocessors. These coprocessors plug into NuBus expansion slots within the Mac in order to provide far greater computational power to solve computationally intensive vision and image analysis problems. For example, currently available coprocessors for the Mac II include: TI MicroExplorer and Symbolics MacIvory LISP machines on a board (board plus basic software is about $10K apiece), the MacDSP board by Spectral Innovations, Santa Clara, CA has a 24MFlop DSP (at about $3K), the Levco transputer board, the Perceptics image processor, and others. As with all coprocessors in the current state-of-the-art and market, integration can often be difficult, though some coprocessors are easier to integrate than others. For example, the LISP machines provide fairly simple mechanisms for subtasking via remote processor calls (RPCs), and they have their own virtual memories (something not found in most other coprocessors on the market).

In general, the parallel computing and Mac coprocessor markets and products are just now emerging. The quality and software support for these coprocessors is rapidly increasing, while their cost is decreasing. The recent rapid development of DSP chips with over 50MFlops performance is sure to impact the power of these coprocessor boards. Apple is currently introducing an extension to their operating system called MR-DOS which provides consistent mechanisms for multitasking, interprocess communication, priority scheduling and timer services and configurability. This is intended to produce a well-defined, consistent software interface for future coprocessors developed for the Mac. It is expected that these new coprocessors will greatly extend processor speed, size and speed of on-board memory, bus architecture bandwidth, the generality of applications which can run on the coprocessor, ease of use, programmability, and the quality of their development environment.

2.2.5 Extended User Interface Support:

The basic sensory modalities of the user interface include visual (via the display), touch (via the keyboard and mouse), and sound (via the built-in Mac stereo sound and voice generation capabilities). Other sensory modalities may be added using off-the-shelf components to expand the ease, power, and naturalness with which the user may interact with the IU environment. For example, the interface may be expanded to include voice keyword recognition (e.g., the VoiceNavigator by Articulate Systems of Berkeley, CA retails for about $750 and operates without application software modification), a data glove (soon to be available by

Nintendo at relatively low cost), 6-d trackball (e.g., from CIS Graphics, Westford, MA for $3300 retail), a touch screen (e.g., from MicroTouch Systems, Woburn, MA retail for $900 and up), and other sensory input devices. These additional sensory modalities expand the potential application domains, ease of use, and user expressiveness over the base level system.

2.2.6 Hardcopy Capabilities:

It is important that the results of processing be transferred into hardcopy for the purposes of reporting results (e.g., final reports, published papers), transferring data (e.g., text files, images), and human-readable archival. Because the Mac is popular in the desktop publishing market, color printers, high resolution laser printers, slide and photo generators, and other hardcopy generation devices are widely available and competitively priced. If volume is not high enough to justify the purchase of more expensive equipment (e.g., a color printer), service bureaus are widely available to obtain hardcopy from Mac II datafiles.

2.3 CURRENT SYSTEM

An extended IU environment workstation has been built at ADS. The workstation is intended primarily for computer vision research and image analysis. These domains typically require more advanced capabilities than afforded by the base level system, so modular extensions have been added. As shown in Figure 3, these extensions include a 600MB magneto optical disk with removable media, an additional hi-resolution color monitor and video card, hi-resolution hardcopy capabilities, and extensive video image acquisition and digitizing facilities (see Figure 4). Coprocessors have also been explored as a means to expand the computational power of the platform (e.g., we evaluated the TI MicroExplorer and Symbolics MacIvory LISP coprocessor boards). These modular extensions have provided us with a powerful and flexible environment for operating within the vision and image processing *domains*.

In addition to the base level system, we have added a RasterOps 19" color monitor (model 1948S) which retails for about $5K and a SuperMac Spectrum/8 video card (1024 x 768 x 8 bits) which retails for about $1500.

A SCSI connected Jasmin 600MB magneto optical removable drive was added (it uses the new Ricoh drive). Each cartridge holds 300MB on each side, it has an average access time of 50ms (about twice the time required for most magnetic drives, though writing on the optical media takes longer), and a 1.2MB data transfer rate. The drive retails for about $5K and each removable cartridge retails for about $270. For tape backups, we transfer data over the ethernet to our timeshare machines which have attached 9-track and 8mm tape drives.

Figure 4 shows the video image acquisition subsystem we have added to the base level system. NTSC video sources are provided by a Sony LDP-1500 videodisk player, 3/4" videotape (Sony VO5800 recorder, VO5850 Recorder, and RM440 remote editing deck), and a Sony color broadcast quality camera. A FOR-A digital time-based corrector reduces image noise by providing better video stability and freeze-frame capability (an especially important consideration for videotape). The videotape editing deck provides a desktop videotape controller which is placed near the operator console to allow convenient videotape and image access control. Because these decks do not have good freeze frame and random access abilities, they are not well-suited for the digitization of imagery in a motion sequence. The videodisk player and direct camera input has proven to be better for motion sequence digitization. The videodisk player is controlled by the Mac through its RS-422/232 port using the Voyager Videostack by The Voyager Company, Santa Monica, CA (about $60)

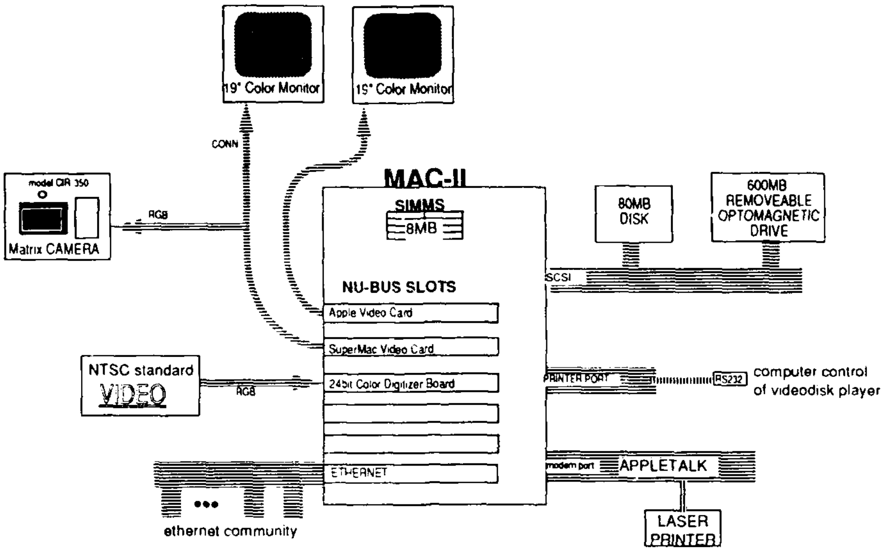


Figure 3: Current Workstation Configuration

which allows simple computer control of playback which includes forward play, fast play, single step, random frame jumps, freeze frame, rapid scan, disk eject and close, frame display, and unit reset. We plan to build a simple port interface to control the videodisk player via user program control and from within CommonLISP. Video data is displayed on two Sony video monitors.

We currently use an 8-bit monochrome Data Translation (DT) video digitizer board with four software-selectable input video channels (which we use as composite color, red, green, and blue) which retails for about $1100. The digitizer captures 640 x 480 images in 1/30th second. Color images may be captured by digitizing and saving the color planes in sequence. We currently drive the digitizer from the interactive program supplied by DT, and we also have a Pascal/C driver which allows direct capture of images from the digitizer into our preferred CVL image format. We eventually plan to integrate the current program interface to allow users to drive the digitizer functions from CommonLISP.

2.4 FUTURE MAC-BASED SYSTEMS

Several changes are currently underway in the Mac and general computing environment which will greatly impact the modular extensions available for the base level system.

Storage and archival technologies are rapidly developing faster, larger, and less expensive memories. The availability of large, low-cost secondary memories provides greater latitude in the computation/power tradeoff for IU environments, research, and applications development. Magneto optical media and 8mm tape systems will most rapidly develop in the near future. The recent advent of DMA boards has overcome some limitations in the Mac II systems (e.g., slow SCSI chip set, no DMA) which will result in far lower average

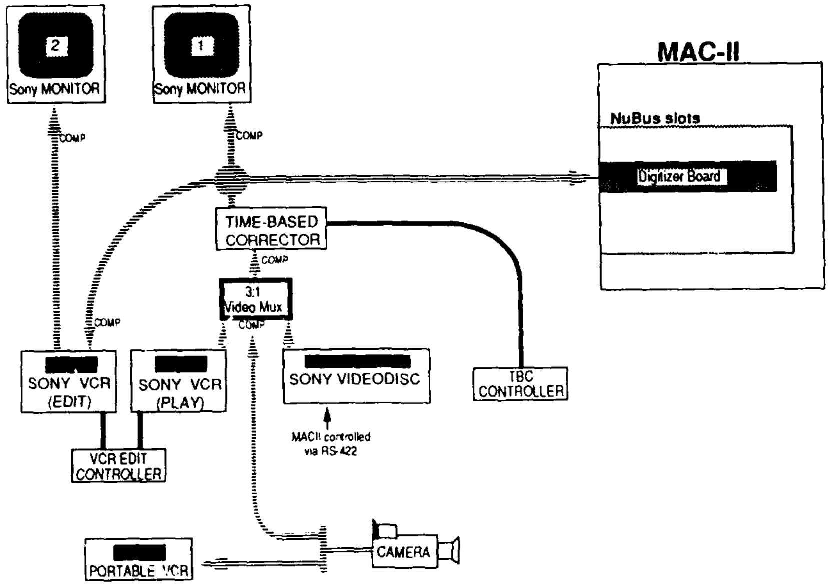


Figure 4: Video Image Acquisition Subsystem

access times for these secondary devices.

Coprocessors for the Mac and parallel processing in general is rapidly evolving. More parallel coprocessors are becoming available for the Mac, and the use of these coprocessors allows the overall computational power of the system to be stated in terms of hundreds of MIPS (versus the 3MIPS in the Mac x-series machines). In this framework, the Mac is primarily a chassis and graphics front-end processor. The recent development by Apple of MR-DOS is intended to standardize the communication and tasking in such an arrangement, and this standard may help to accelerate the development of these systems. The rumored addition by Apple of an AMD 29000 RISC graphics coprocessor to the Mac (supposedly to be introduced in late '89 or early '90) will offload graphics tasks from the main MAC CPU and significantly boost overall graphics and processing power.

Image acquisition and display technologies are also rapidly developing. It appears likely that within the next year or so, real-time digitizer to large optical media storage will be commercially available. Such systems are also capable of scanning digital data to video in real-time. This would provide high quality, random access, digital image acquisition, and real-time display of processing results at relatively low cost; current hard disk systems which currently do this run into the six digits. The ability to redisplay processing results in real-time which may have taken far longer to process (e.g., motion processing or segmentation results) will greatly help to describe algorithms and results in the community.

3. CONCLUSION

In general, we found the Mac to be a very good graphical platform which facilitated the rapid prototype development of an IU environment on a low-cost, modularly expandable system. The base Mac system without the addition of coprocessor boards did not have sufficient computational power to support moderate to high computational requirements which usually occur in advanced IU applications. The use of coprocessor boards was explored and found to be a viable way to greatly increase the power of the base system to better handle increased computational loads. Upcoming improvements to the Mac operating system to handle virtual memory, multiprocessing, and larger memories will make the Mac better suited for IU environments and applications.

ACKNOWLEDGMENTS

This document was prepared by Advanced Decision Systems (ADS) of Mountain View, California, under U.S. Government contract number DACA76-85-C-0005 for the U.S. Army Engineer Topographic Laboratories (ETL), Fort Belvoir, Virginia, and the Defense Advanced Research Projects Agency (DARPA), Arlington, Virginia. The authors wish to thank Randy Walser, Tom Binford, and Kevin Willey for their contributions and assistance.

References

[Apple Computer - 88] Apple Computer, *Inside Macintosh*, Vols. I-V, Addison-Wesley Publ., Reading, Massachusetts, 1988.

[Edelson, et.al. - 88] D. Edelson, J. Dye, T. Esselman, M. Black, and C. McConnell, "VIEW Programmer's Manual", Advanced Decision Systems, Mountain View, California, June, 1988.

[Lawton and McConnell - 88] D. Lawton and C. McConnell, "Image Understanding Environments", in *IEEE Proceedings*, August, 1988.

[McConnell et.al. - 87] C. McConnell, P. Nelson and D. Lawton, "Constructs for Cooperative Image Understanding Environments", *Proceedings of the DARPA Image Understanding Workshop*, Los Angeles, California, February, 1987.

[Riley et.al. - 87] K. Riley, C. McConnell, and D. Lawton, "Powervision", Advanced Decision Systems, Mountain View, California, April, 1987.

CENTRALIZED AND DECENTRALIZED KALMAN FILTER TECHNIQUES FOR TRACKING, NAVIGATION, AND CONTROL

Christopher Brown
Computer Science Department
University of Rochester
Rochester
NY 14627

Hugh Durrant-Whyte
John Leonard
Bobby Rao
Robotics Research Group
Department of Engineering Science
Oxford OX1 3PJ

ABSTRACT

A review of some estimation basics is followed by an illustrative application of the variable dimension Kalman filter for tracking a maneuvering target. The performance of the nearest neighbor standard filter is compared to that of the probabilistic data association filter for tracking a target in clutter. Multi-target tracking, using sonar sensors to estimate an autonomous robot's distance from walls, is applied to the navigation problem. The Kalman filter equations can be completely decentralized and distributed among the nodes of a multi-sensor system. Each sensing node implements its own local Kalman filter, arrives at a partial decision, and broadcasts it to every other node. Each node then assimilates this received information to arrive at its own local but optimal estimate of the system state.

ESTIMATION AND KALMAN FILTERING

ESTIMATION

Non-Bayesian Estimation

Non-Bayesian Estimation is used when the value being estimated is not a random variable but is *constant*. The estimate should converge to this value as the number of readings $\longrightarrow \infty$.

Assuming the prior Probability Density Function (PDF) of x is unknown, its posterior PDF is unavailable, leading to the use of a likelihood function:

$$\lambda_k(x) = p(z^k|x)$$

and hence the Maximum Likelihood (ML) method

$$\hat{x}(k) = argmax_x p(z^k|x).$$

The likelihoods arise from empirical or analytic models of the sensor.

Bayesian Estimation

Bayesian Estimation is used when the parameter to be found is a random variable (RV) with a PDF $p(x)$. A value of x is assumed to have occurred via this PDF and remained constant during its measurement sequence. The measurement sequence should converge to the actual value of x that we are measuring.

Given the prior PDF for x, its posterior PDF follows from Bayes' Rule

$$p(x|z^k) = \frac{p(z^k|x)p(x)}{p(z^k)}.$$

This leads to the Maximum A Posteriori (MAP) method:

$$\hat{x}(k) = argmax_x p(x|z^k) = argmax_x[p(z^k|x)p(x)].$$

Both the ML and MAP methods yield modes of a probability distribution (the most likely value).

Least Squares Estimation

The LS method is for non-random parameters and for measurements

$$z(j) = h(j,x) + w(j),$$

which yields

$$\hat{x}(k) = argmin_x \sum_{j=1}^{k} [z(j) - h(j,x)]^2.$$

Here $h(.)$ is the sensor model. The least squares techniques yields the mean, not the mode, of the probability distribution function.

Minimum Mean-Square Error Estimation

The MMSE method is for random parameters and yields

$$\hat{x}(k) = argmin_{\hat{x}} E[(\hat{x} - x)^2|z^k].$$

Linear Estimation

The estimator is a linear function of the measurement data.

$$\hat{x} = \bar{x} + P_{xz}P_{zz}^{-1}(z - \bar{z})$$

where

$$P_{xx} = E[(x - \bar{x})(x - \bar{x})^t],$$

$\bar{z}$ is the expected measurement, and $(z - \bar{z})$ is the *innovation* produced by the true measurement.

Equivalences Of Methods

All the above can produce the same results under certain conditions:

- ML ≡ MAP

  This occurs when the prior PDF for x is non-informative, ie when $p(x) = \epsilon$ and $\epsilon \longrightarrow \infty$ or when the PDF is a Gaussian with $\sigma_0 \longrightarrow \infty$

- LS ≡ ML

  This occurs if the noises $w(j)$ in $z(j) = h(j,x) + w(j)$ are independent identically distributed (IID) zero mean Gaussian RVs. This also applies if x is a vector property.

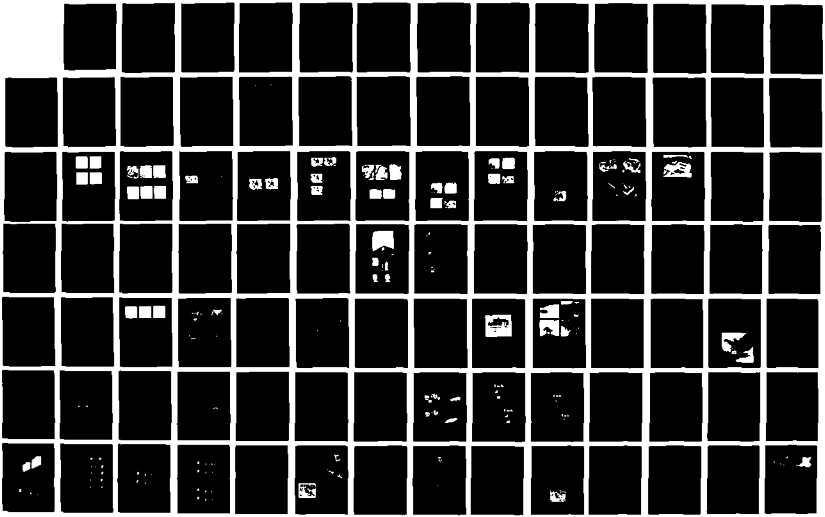
- MMSE ≡ MAP

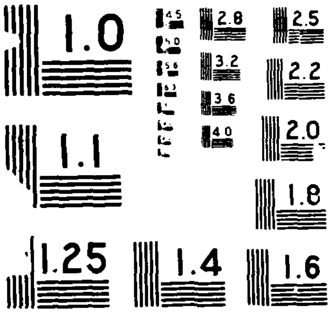
  These coincide if the posterior PDF of x is Gaussian with arbitrary mean.

- Linear Estimation

  When x is a Gaussian RV the linear estimator of the MMSE form is equivalent to the best *linear* estimator for arbitrarily distributed RVs with the same first and second moments. However if the RVs are not Gaussian the MMSE can be a better estimator than the linear estimator (ie. Gaussian RVs give the worst case results of an MMSE estimator).

AD-A212 981   PROCEEDINGS OF IMAGE UNDERSTANDING WORKSHOP HELD IN   8/12
PALO ALTO CALIFORNIA O.. (U) SCIENCE APPLICATIONS
INTERNATIONAL CORP MCLEAN VA L S BAUMAN MAY 89
UNCLASSIFIED   N00014-86-C-0700   F/G 17/11   NL




1.0
4.5
5.0
5.6
2.8
3.2
3.6
4.0
2.5
2.2
2.0
1.1
1.8
1.25
1.4
1.6

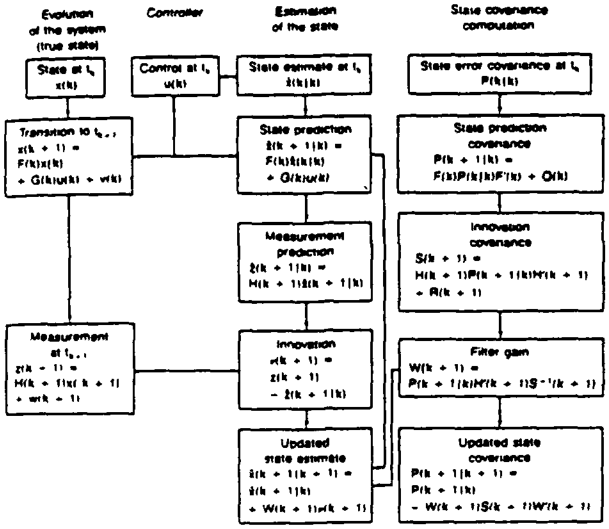

Evolution of the system (true state)
Controller
Estimation of the state
State covariance computation
State at t_k x(k)
Control at t_k u(k)
State estimate at t_k x̂(k|k)
State error covariance at t_k P(k|k)
Transition to t_{k+1} x(k + 1) = F(k)x(k) + G(k)u(k) + v(k)
State prediction x̂(k + 1|k) = F(k)x̂(k|k) + G(k)u(k)
State prediction covariance P(k + 1|k) = F(k)P(k|k)F'(k) + Q(k)
Measurement prediction ẑ(k + 1|k) = H(k + 1)x̂(k + 1|k)
Innovation covariance S(k + 1) = H(k + 1)P(k + 1|k)H'(k + 1) + R(k + 1)
Measurement at t_{k+1} z(k + 1) = H(k + 1)x(k + 1) + w(k + 1)
Innovation ν(k + 1) = z(k + 1) − ẑ(k + 1|k)
Filter gain W(k + 1) = P(k + 1|k)H'(k + 1)S^{-1}(k + 1)
Updated state estimate x̂(k + 1|k + 1) = x̂(k + 1|k) + W(k + 1)ν(k + 1)
Updated state covariance P(k + 1|k + 1) = P(k + 1|k) − W(k + 1)S(k + 1)W'(k + 1)

Figure 1: The (Linear) Kalman Filter (one cycle).

THE KALMAN FILTER

Kalman filtering is a form of optimal estimation characterized by recursive (i.e. incremental) evaluation, an internal model of the dynamics of the system being estimated, and a dynamic weighting of incoming evidence with ongoing expectation that produces estimates of the state of the observed system. The basic Kalman filter loop appears in Fig. 1 (taken from [BF88]). Its input is the system measurements, its apriori information is the system dynamics and noise properties of system and measurement, and its useful outputs are the *innovation* (the difference between the predicted and observed measurement, by which the filter's performance may be quantified), and the estimated system state.

Consider a system with state vector **x**, taking observations **z** with Gaussian noise sources **w** and **v** respectively:

$$\mathbf{x}(k+1) = \mathbf{F}(k)\mathbf{x}(k) + \mathbf{G}(k)\mathbf{w}(k) \tag{1}$$

$$\mathbf{z}(k) = \mathbf{H}(k)\mathbf{x}(k) + \mathbf{v}(k) \tag{2}$$

where

$$E\left\{ \begin{pmatrix} \mathbf{w}(k) \\ \mathbf{v}(k) \end{pmatrix} \begin{pmatrix} \mathbf{w}^T(j), \mathbf{v}^T(j) \end{pmatrix} \right\} = \begin{pmatrix} \mathbf{Q}(k) & \mathbf{S}(k) \\ \mathbf{S}^T(k) & \mathbf{R}(k) \end{pmatrix} \delta_{k,j} \tag{3}$$

For such a system the conventional Kalman Filter equations for state prediction $\mathbf{x}(k+1 \mid k)$, variance prediction $\mathbf{P}(k+1 \mid k)$, state update $\mathbf{x}(k+1 \mid k+1)$ and variance update $\mathbf{P}(k+1 \mid k+1)$, where the variance is defined as

$$\mathbf{P}(k \mid k) = E\{[\mathbf{x}(k) - \hat{\mathbf{x}}(k \mid k)][\mathbf{x}(k) - \hat{\mathbf{x}}(k \mid k)]^T \mid \mathbf{Z}^k\}$$

may be found in (Bar-Shalom [BF88]) and have the following form:

Prediction:

$$\hat{\mathbf{x}}(k+1 \mid k) = \bar{\mathbf{F}}\hat{\mathbf{x}}(k \mid k) + \mathbf{GSR}^{-1}\mathbf{z} \tag{4}$$

$$\mathbf{P}(k+1 \mid k) = \bar{\mathbf{F}}\mathbf{P}(k \mid k)\bar{\mathbf{F}}^T + \mathbf{G}\bar{\mathbf{Q}}\mathbf{G}^T \tag{5}$$

Update:

$$\hat{\mathbf{x}}(k+1 \mid k+1) = [\mathbf{I} - \mathbf{KH}]\hat{\mathbf{x}}(k+1 \mid k) + \mathbf{Kz} \tag{6}$$

$$\mathbf{P}^{-1}(k+1 \mid k+1) = \mathbf{P}^{-1}(k+1 \mid k) + \mathbf{H}^T\mathbf{R}^{-1}\mathbf{H} \tag{7}$$

where

$$\mathbf{K} = \mathbf{P}(k+1 \mid k+1)\mathbf{H}^T\mathbf{R}^{-1} \tag{8}$$

$$\bar{\mathbf{F}} = \mathbf{F} - \mathbf{GSR}^{-1}\mathbf{H} \tag{9}$$

The (first order) extended Kalman filter (EKF) is a version of the Kalman filter that deals with nonlinear dynamics or nonlinear measurement equations, or both. It linearizes the problem around the predicted state (a second-order EKF makes a second-order approximation). The basic control loop of Fig. 1 still applies, but measurements are predicted using the nonlinear measurement equation $\mathbf{h}(.)$. The measurement model $\mathbf{h}[k, \mathbf{x}(k)]$ is linearized about the current *predicted* state vector, $\hat{\mathbf{x}}(k+1 \mid k)$ using the Jacobian of the nonlinear measurement function $\mathbf{h}(.)$. The calculations for filter gain, state update, and covariance update use the Jacobian $\mathbf{h}_\mathbf{x}(.)$. Likewise state prediction is accomplished using the nonlinear state equation $\mathbf{f}(.)$. The plant model $\mathbf{f}[k, \mathbf{x}(k)]$ is linearized about the current *estimated* state vector, $\hat{\mathbf{x}}(k \mid k)$. The state prediction covariance is computed using the plant Jacobian $\mathbf{f}_\mathbf{x}(k)$.

For ease of notation, we use the measurement Jacobian to define the measurement prediction covariance matrix $\mathbf{S}(k+1)$:

$$\mathbf{S}(k+1) = \mathbf{h}_\mathbf{x}(k+1)\mathbf{P}(k+1 \mid k)\mathbf{h}_\mathbf{x}{}^T(k+1) + \mathbf{R}(k+1) \tag{10}$$

With these definitions of $\mathbf{f}_\mathbf{x}(k)$, $\mathbf{h}_\mathbf{x}(k+1)$, and $\mathbf{S}(k+1)$, the extended Kalman filter equations are the following.

$$\hat{\mathbf{x}}(k+1 \mid k) = \mathbf{f}[k, \hat{\mathbf{x}}(k \mid k)] \tag{11}$$

$$\mathbf{P}(k+1 \mid k) = \mathbf{f}_\mathbf{x}(k)\mathbf{P}(k \mid k)\mathbf{f}_\mathbf{x}{}^T(k) \tag{12}$$

$$\mathbf{K}(k+1) = \mathbf{P}(k+1 \mid k)\mathbf{h}_\mathbf{x}(k+1)\mathbf{S}^{-1}(k+1) \tag{13}$$

$$\hat{\mathbf{x}}(k+1 \mid k+1) = \hat{\mathbf{x}}(k+1 \mid k) + \mathbf{K}(k+1)[\mathbf{z}(k+1) - \mathbf{h}[k, \hat{\mathbf{x}}(k+1 \mid k)]] \tag{14}$$

$$\mathbf{P}(k+1 \mid k+1) = \mathbf{P}(k+1 \mid k) - \mathbf{K}(k+1)\mathbf{S}(k+1)\mathbf{W}^T(k+1) \tag{15}$$

In the formulation above and in the examples of the next section, one process receives all measurements and makes the estimation. There has been a considerable amount of recent interest in the development of algorithms to process information obtained from a distributed sensor system. This interest arises from a need to use parallel processing architectures efficiently. Parallel computation is needed if multi-sensor systems are to be able to process their data in real-time. A decentralized filter is presented below.

CENTRALIZED FILTERS

MANEUVERING TARGETS

When targets *maneuver*, i.e. depart from the basic, steady-state, "normal" dynamic behaviour, a tracking filter must respond. To the filter, maneuvering is signaled by a rapid increase in the normalized innovation. Recommended methods for dealing with this situation include the following.

1. Increase the process noise, or certain components of it, attributed to the target.
2. Use several filters with different assumptions in parallel, and combine their outputs probabilistically.
3. Create new filters as needed, pursuing a hypothesis tree of parallel hypotheses about target state. This tree must be pruned rapidly lest its maintenance overwhelm the computational resources.
4. Model maneuvers as colored (correlated) noise: in particular model target acceleration as a zero-mean, first-order Markov process (one with exponential autocorrelation).
5. Perform *input estimation*, in which measurements based on the nonmaneuvering model are used to detect and estimate the control input applied to the plant dynamics, and that control input in turn is used to correct the state estimate.
6. Use *variable dimension filtering*, in which the maneuver is considered part of the plant dynamics, not noise. Maneuver detection causes the substitution of a different, higher-order dynamic model for the lower-order, "quiescent" model.

Bar-Shalom [BF88] compares the performance on maneuvering targets of three filters: two-level white noise, variable dimension (VD) filtering, and input estimation (IE). He notes that the computational effort ratios between the three are 1:2:8. His study reveals that the two-level white noise filter does surprisingly well, being slightly worse than the VD filter but definitely better than the IE filter.

We chose to implement the VD filter, in the light of its relatively low computational cost and relatively high efficacy in the Bar-Shalom study. Our illustrative application was to a target moving in two dimensions at constant velocity until some time at which it begins constant acceleration in the same direction. The quiescent filter is simply the constant-velocity target filter, the maneuvering filter is for a constant acceleration target.

Fig. 2(a) shows the normalized innovation (i.e. error measure) of the constant-velocity filter as time passes. Performance starts degrading at $T = 5$, when acceleration begins. Fig. 2(b) shows the corresponding estimate of y position, which gradually degrades through time. The VD filter has remarkably better performance,

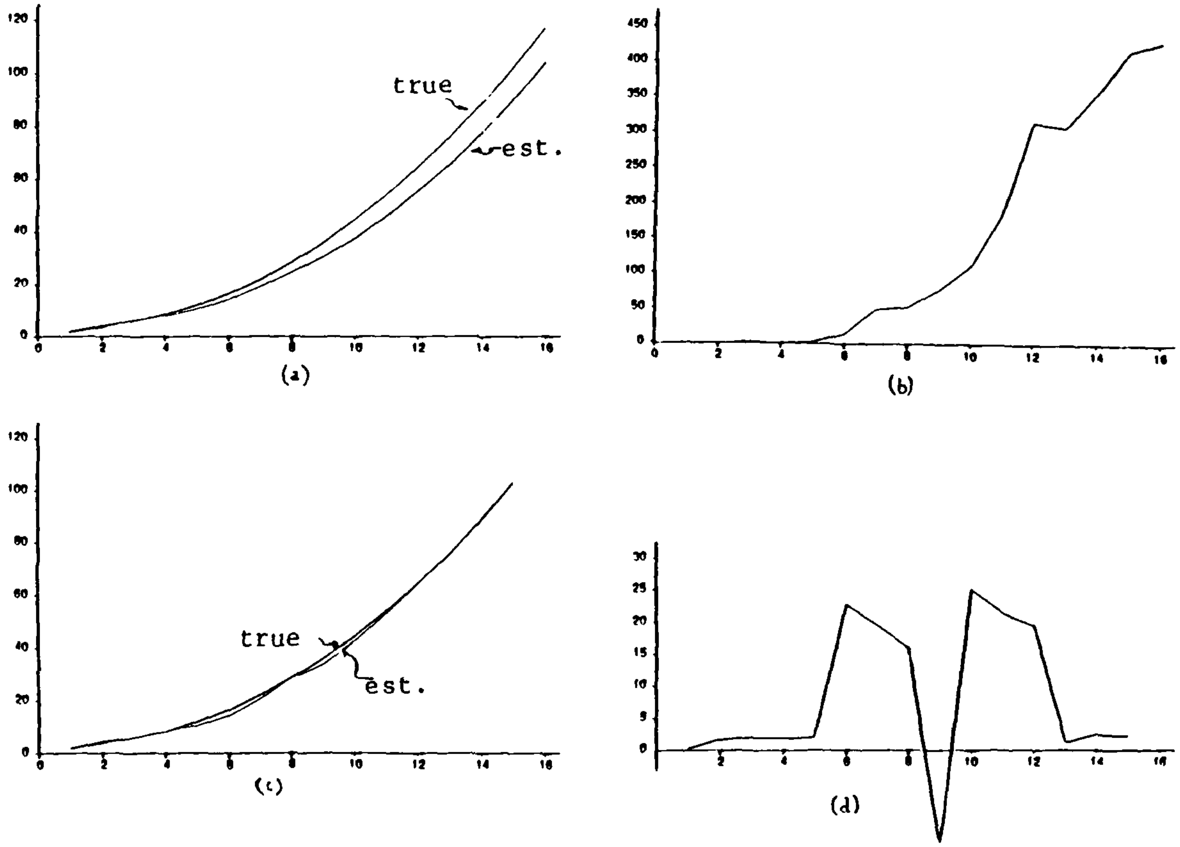


Figure 2: Performance of the Variable Dimension filter. (a) Estimated y position .ur target that starts maneuvering at T=5. (b) Normalized innovation of quiescent filter applied to target. (c) Estimated y position for target from VD filter. (d) Normalized innovation of VD filter (note scale change compared to (b)).

which underlines the importance of accurate dynamic modeling. Fig. 2(c) shows the normalized innovation of the VD filter (note the scale change). The maneuvering filter was switched in at $T = 6$. Fig. 2(d) shows the corresponding estimate of y position.

TARGETS IN NOISE

Tracking an object in the presence of spurious measurements (*clutter*) can be done in several ways. All assume a *validation gate* outside of which measurements are ignored: its size is a function of the desired probability of including the true measurement, and can be derived from a chi-squared calculation applied to the normalized innovation.

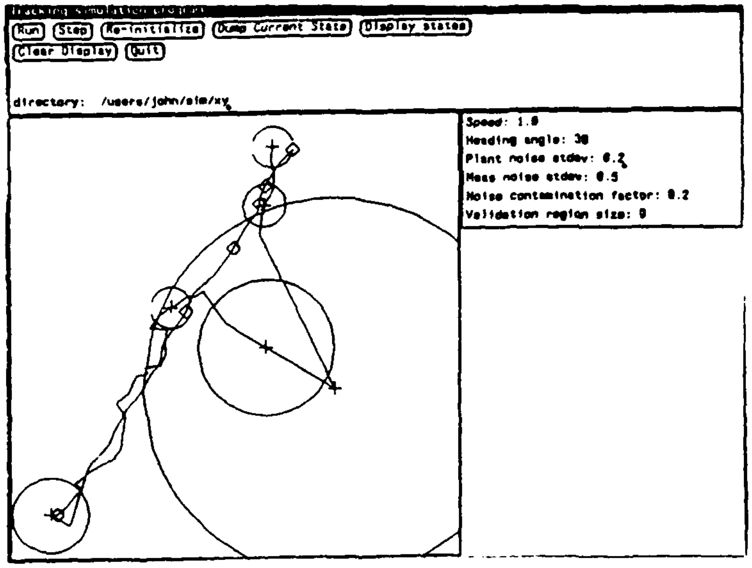
1. The optimal way to track a single target in clutter is the *track-splitting* approach, in which a tree of possible tracks is maintained. This can be a combinatorially expensive method.
2. An obvious alternative to treating all measurements, as it were, in parallel, is to pick a single candidate measurement and proceed as if it were the right one. The obvious candidate is the one closest in measurement space (that one with smallest normalized innovation), so this technique is called the *nearest-neighbor standard filter* (NNSF). The problem is that the true measurement can be missed.
3. A third approach is the *probabilistic data association filter*, or PDAF. In it, the measurements within the validation gate are probabilistically blended to yield a combined innovation which is input into the Kalman filtering process. The problem is that the result does not correspond to that of any actual measurement.

The application illustrated in Figure 3 is tracking a constant-velocity target moving in two dimensions. The plant and measurement are both noisy, with the measurement noise being drawn from a contaminated Gaussian, in which with some probability the measurement noise has different parameters (here, higher variance) than the noise expected by the filter. The filter produces a validation gate, based on the innovation covariance. A measurement within the validation gate is used, while one outside the gate is ignored. In the run illustrated, the elliptical validation gate (here a circle) shrinks while good measurements are obtained, and grows when a measurement is missed — this adjustment makes reacquisition more likely. The figure shows snapshots of the validation gate during a serious loss and reacquisition of track.

General aspects of the behaviour of the NNSF and PDAF filters may be predictable on abstract grounds. For instance we might make the following predictions for uniformly distributed clutter and high *probability of detection* (probability that the target is detected at all, either inside or outside the validation gate.)

1. With both the NNSF and PDAF filters tracking in clutter, as time goes on it is increasingly likely that the filter will "lose track", e.g. start tracking clutter, or have an estimate of target state outside some fixed bound.
2. With a "non-maneuvering" NNSF filter, low and high clutter levels may be less immediately harmful than medium levels, since with low clutter the target is likely to be nearest the predicted state, and with high clutter there is likely to be a clutter point near the predicted state. It would seem that at intermediate level the clutter would be more likely to attract the filter away from the target.
3. The NNSF filter would seem more likely to make serious errors by tracking clutter since it does not weigh the evidence. The performance of the PDAF should degrade more gracefully as conditions get worse.

We implemented the NNSF and PDAF filters, and used them to provide individual output tracks, as in the previous work. Also the programs were embedded in Monte Carlo simulations to provide data over a number of runs in statistically similar situations. The results confirm the above expectations but also provide



directory: /users/john/sim/xy

Speed: 1.0
Heading angle: 30
Plant noise stdev: 0.2
Meas noise stdev: 0.5
Noise contamination factor: 0.2
Validation region size: 9

Figure 3: Real (diamond) and estimated (cross) target tracks, and elliptical validation gate, through time, showing loss and reacquisition of track.

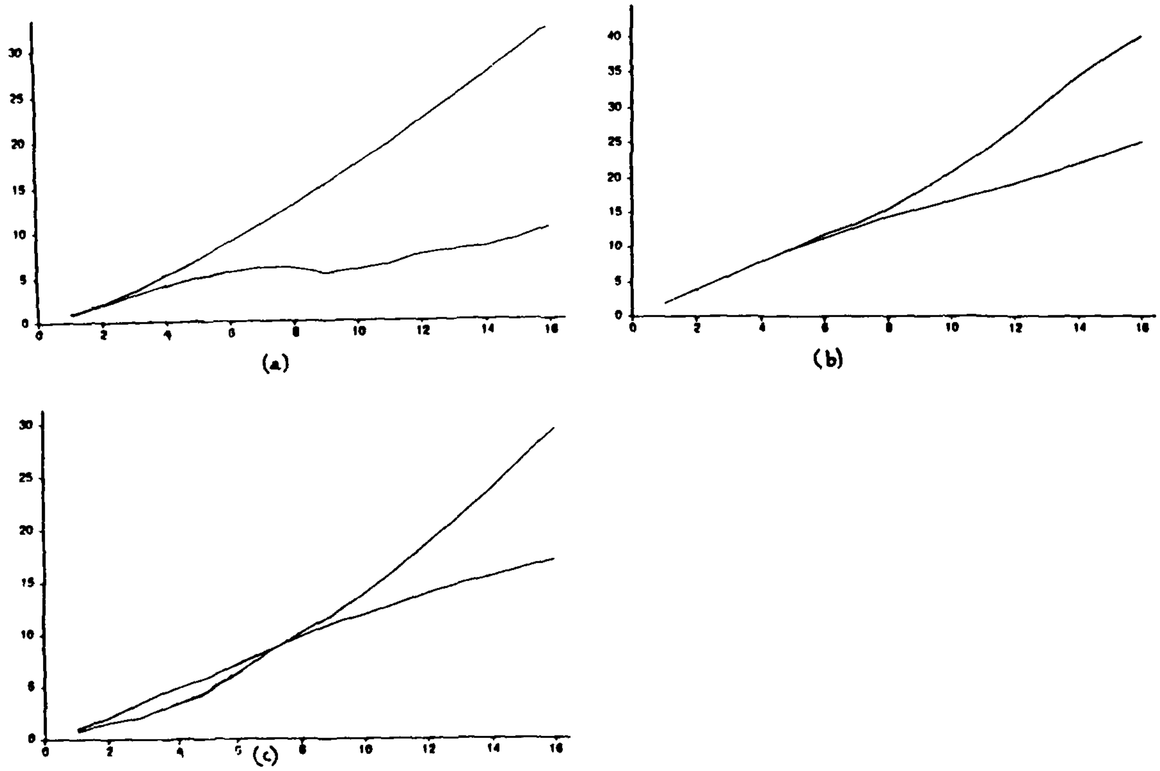


Figure 4: In this figure and the next, the filter is a constant-velocity (linear) Kalman filter. The plant noise is an acceleration component (white noise of mean 0.0 and variance $q = 0.2$.) Clutter density $\sigma = 0.4$. (Parallelepidal) validation gate size is such that .99 of target measurements should fall within it initially. The measurement noise has variance 0.2 for the x, 0.1 for y. (a) Error in X vs time in NNSF, tracking situation parameters as in text, clutter density 0.5. (b) Error in Y vs time in NNSF, clutter density 1.0. (c) Error in X vs time in NNSF, clutter density 2.0.

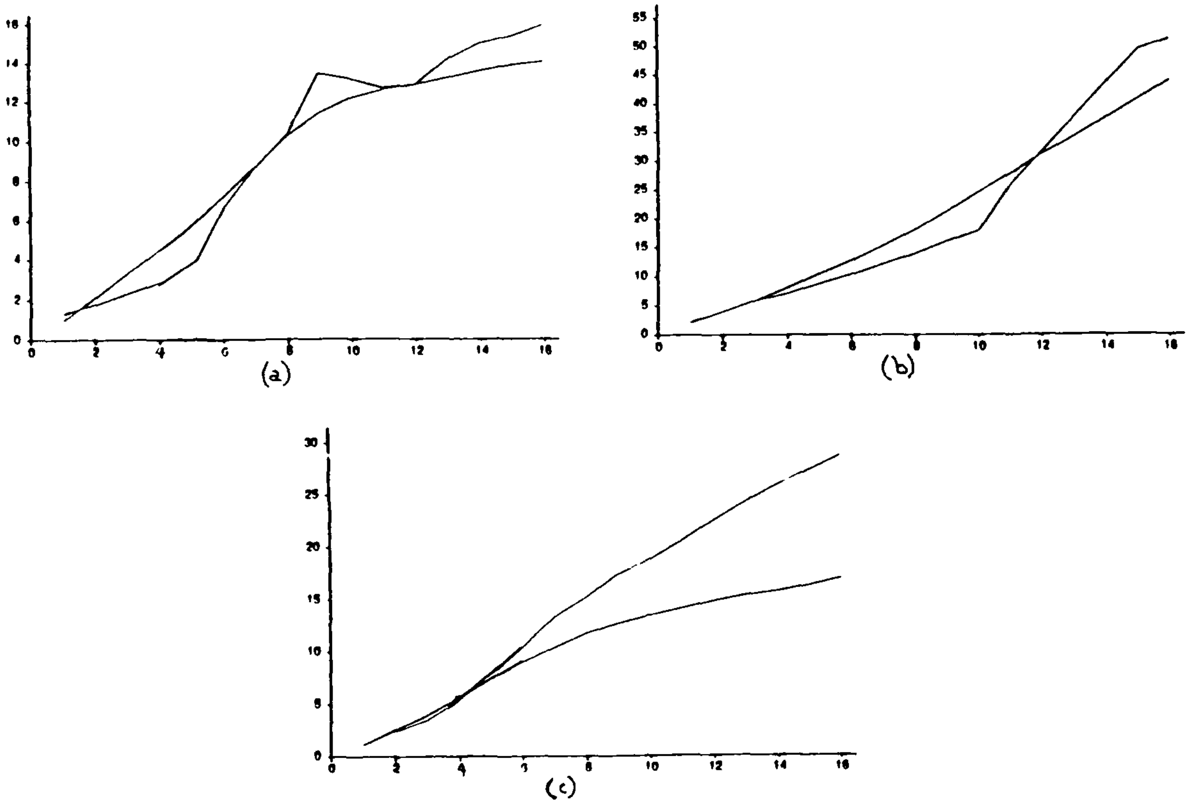


Figure 5: (a) Error in X vs time in PDAF, tracking situation parameters as in text, clutter density 0.5. (b) Error in Y vs time in PDAF, clutter density 1.0. (c) Error in X vs time in PDAF, clutter density 2.0.

some surprises. Fig. 4 shows individual tracking runs for situations with uniformly distributed clutter of increasing density. The number of clutter points in the validation gate was determined by rounding a normal variate of indicated mean (the clutter density) and standard deviation to the nearest integer, and uniformly distributing the resulting number of clutter points throughout the validation gate volume. In these runs the volume was close to unity. The NNSF figures should be compared with Fig. 5, which shows PDAF results for similar situations.

Figs. 4 and 5 illustrate indeed that lower clutter levels can result in worse NNSF performance than higher levels, and that performance of both filters falls off as time increases. They perhaps furnish a mild surprise, *viz.* the ability of the PDAF to reacquire the track. This happens for lower clutter levels, and presumably occurs when the "signal to noise" ratio is high in the validation gate (e.g. there are few measurements in the validation gate, and at least one of them is near the actual position of the target and correctly is accorded high weight). The behavior of the PDAF in high clutter conditions is not as surprising — it drifts, taking the average of the random clutter.

Fig. 6 shows statistics gathered over $N = 50$ runs with the NNSF filter. It should be compared to Fig. 7. The plots show the fraction of *lost tracks* and the *average final error* of the filter's estimate. Both functions vary over the set of tracking times {4, 8, 16, 32 } timesteps, and both vary over the set of clutter densities { 0.25, 0.5, 1.0, 1.5, 2.0, 3.0 }. Here "lost track" is defined as the estimated position being more than some fixed distance (here 2.0) from the actual position of the target. Thus it is possible to imagine the filter actually tracking clutter for the entire run, going wildly wrong in its estimates, but luckily arriving inside the threshold distance just at the last step, and thus not "losing track" by this definition.

The average final error function is meant to quantify the filter performance more than the discrete "lost

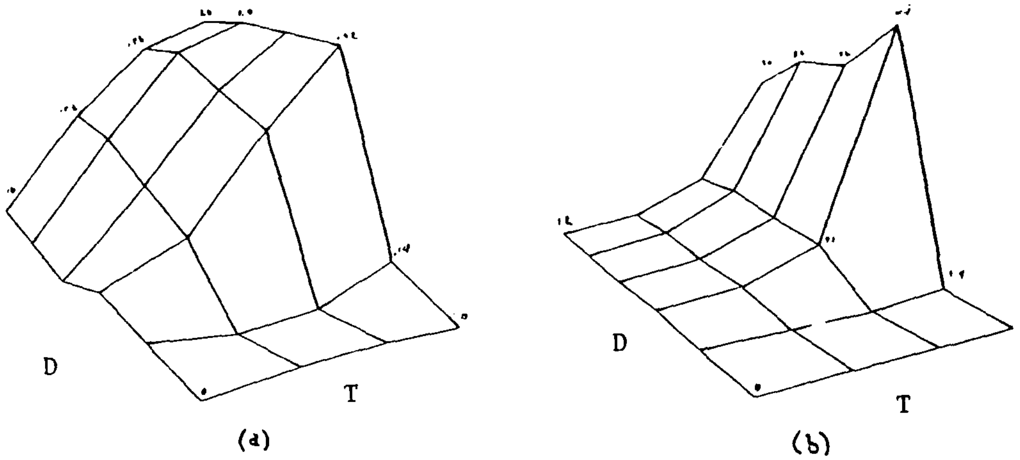


Figure 6: This figure and the next show the fraction of *lost tracks* (a) and the *average final error* (b) of the filter's estimate. Both functions vary over the set of tracking times {4, 8, 16, 32 } timesteps (axis T), and both vary over the set of clutter densities { 0.25, 0.5, 1.0, 1.5, 2.0, 3.0 } (axis D). "Lost track" means the estimated position is more than some fixed distance from the actual position of the target. This figure shows results for the NNSF, with tracking situation as in previous figures.

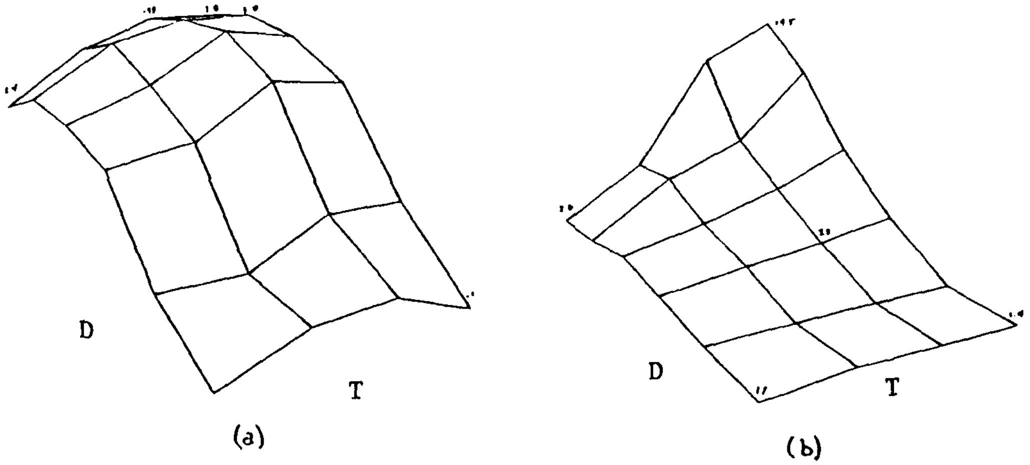


Figure 7: Lost tracks (b) and average final error (b) vs track length and clutter density PDAF, with tracking situation as in last figure.

track" measure, illustrating a linear loss function corresponding to the intuition that an estimate closer to the truth at the final timestep is better. Results like those in Figures 4 and 5, we are probably safe in inferring that a closer final estimate also betokens a better estimate throughout the run.

The plots are perhaps surprising in that by this definition of lost track, the NNSF outperforms the PDAF fairly convincingly over a large range. These results are typical of many we obtained, but it is occasionally possible to engineer situations where the PDAF loses track less often. At least it seems fair to say that the situation is more complicated than it appears from Figure 6.1 of [BF88], which indicates a marked superiority of PDAF along axes whose semantics are not clear from the text (perhaps the original paper gives more details).

The plots are perhaps not surprising in that they accord with the prediction of graceful degradation of the PDAF in terms of the average error metric, using which it convincingly outperforms the NNSF. The higher sensitivity of NNSF to intermediate clutter levels is again demonstrated.

NAVIGATION BY MULTI-TARGET TRACKING

We are using a Kalman-filter-based, multi-target tracking approach to mobile robot navigation. The goal is to get optimal performance with a single sensor through a strategy of *Active Sensor Control*. Ultimately, we shall use a hierarchical control strategy with three layers (*attention, looking, recognition*). In the limited navigational task we undertake first (maneuvering in a known laboratory with unknown obstacles), the recognition aspect is subsumed into the looking aspect.

The navigation problem can be divided into the two distinct tasks of localization, (determining the absolute position of the robot), and obstacle avoidance. In the multi-target tracking framework, localization is a process of *looking* for *expected* events; the sensors are directed to instantiate the locations of beacons, which are reliable, easy-to-sense, natural features of the robot's environment, such as the walls of the room. Precise localization of position can then be obtained by using a target-based [BF88] approach to track these beacons as the robot moves. The task of obstacle avoidance is achieved by a measurement-based [Rei79] tracking technique whose attention is captured by unexplained events in the path of the robot. We show the localization component of this navigation technique using a single sonar sensor mounted on a simple mobile robot with point dynamics. Although the scheme has been run on the robot hardware, the illustrations come from a simulator we are developing that will incorporate detailed sensor and environment models for sonar.

Ultimately this work will incorporate both the "top-down" and "bottom-up" control styles that can be found in perception (see the Discussion section). In the full implementation, the *recognition* layer is complex, involving path-planning, obstacle avoidance, and model learning and maintenance — slow processes of a global nature. The attention layer provides a high-speed interface to the real world.

The System Model

The system dynamics for one of Oxford's mobile vehicles are illustrated in Fig. 8(a). The state of the vehicle at time t_k is described by the state vector $\mathbf{x}(k) = (x_k, y_k, \phi_k, T_k)$. x_k and y_k are the x and y coordinates of the center of the vehicle referenced to a global frame. ϕ_k is the orientation of the vehicle referenced to a global frame. T_k is the speed of the vehicle at time t_k, which represents the linear distance the vehicle will travel during the time interval from t_k to t_{k+1}. The motion of vehicle is such that during each timestep it travels first in a straight line forward from position (x_k, y_k) to position (x_{k+1}, y_{k+1}), then rotates to its final orientation ϕ_{k+1}.

$$\begin{aligned} x_{k+1} &= x_k + T_k \cos\phi_k \\ y_{k+1} &= y_k + T_k \sin\phi_k \end{aligned}$$

The robot motion is controlled by specifying a control input $\mathbf{u}(k) = (0, 0, \Delta\phi_k, \Delta T_k)^T$. $\Delta\phi_k$ controls the final rotation applied after the linear movement forward for the current time-step. This rotation is subject to a random disturbance v_ϕ, modeled as zero-mean Gaussian with variance q_ϕ:

$$\phi_{k+1} = \phi_k + \Delta\phi_k + v_\phi, \quad v_\phi \sim N(0, q_\phi)$$

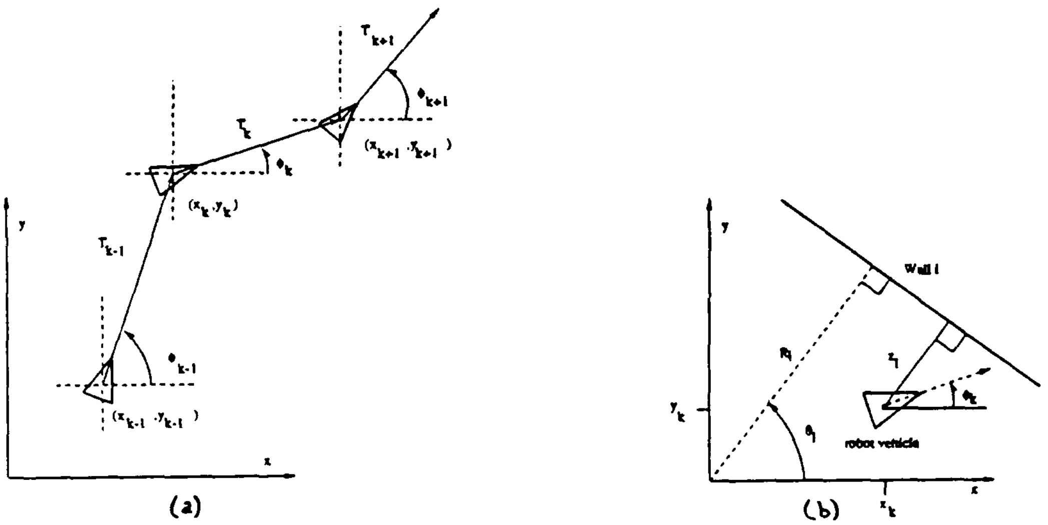


Figure 8: (a) System dynamics for robot vehicle. (b) Taking a range measurement from a wall. (x_k, y_k, ϕ_k) is the position of the robot vehicle in global coordinates. (R_i, θ_i) represents a line in global coordinates. z_i is a noisy measurement of the perpendicular distance from the vehicle to wall $\mathbf{p}_i$.

ΔT_k controls the distance traveled forward for the *next* time-step, which is subject to a random disturbance v_T, modeled as zero-mean Gaussian with variance q_T:

$$T_{k+1} = \Delta T_k + v_T, \quad v_T \sim N(0, q_T)$$

The overall plant equation of the system is

$$\begin{bmatrix} x_{k+1} \\ y_{k+1} \\ \phi_{k+1} \\ T_{k+1} \end{bmatrix} = \begin{bmatrix} x_k + T_k \cos\phi_k \\ y_k + T_k \sin\phi_k \\ \phi_k \\ 0 \end{bmatrix} + \begin{bmatrix} 0 \\ 0 \\ \Delta\phi_k \\ \Delta T_k \end{bmatrix} + \begin{bmatrix} 0 \\ 0 \\ q_\phi \\ q_T \end{bmatrix} \tag{16}$$

which may be written as:

$$\mathbf{x}(k+1) = \mathbf{f}[k, \mathbf{x}(k)] + \mathbf{u}(k) + \mathbf{v}(k), \quad \mathbf{v}(k) \sim N(0, \mathbf{Q}(\mathbf{k})) \tag{17}$$

where $\mathbf{f}[k, \mathbf{x}(k)]$ is a non-linear function of the system state.

The Measurement Model.

The measurement model is for a stationary robot vehicle taking range measurements to a number of walls using a sonar sensor. For now, we assume perfect measurements of ϕ_k, the orientation of the vehicle. (Our system has a 9-bit resolution digital compass that gives us this information. We also have developed reliable algorithms to extract orientation information directly from the sonar data). The model of the environment is a list of n line segments that represent walls and large planar objects in the room. A line segment $\mathbf{p}_i$ is represented by R_i, its perpendicular distance from the origin, and θ_i, its orientation with respect to the origin. In terms of these parameters, the equation of the line is:

$$R_i - x\cos\theta_i - y\sin\theta_i = 0 \tag{18}$$

The sensor returns the minimum distance of any object detected in its 25 degree beam width. Thus if it is pointed in a *direction approximately perpendicular to the wall*, the range value obtained is the perpendicular distance from the vehicle to the wall. The active sensor control framework uses an *a priori* estimate of position $\hat{\mathbf{x}}(k \mid k-1)$ to direct the sensor to look nearly perpendicular to a given wall $\mathbf{p}_i = (R_i, \theta_i)$ to obtain a measurement z_i of the perpendicular distance to the wall (Fig. 8(b)).

Let r_i denote the true perpendicular distance from a given vehicle location (x_k, y_k) to the wall. Then

$$r_i = |R_i - x_k \cos\theta_i - y_k \sin\theta_i|. \tag{19}$$

The absolute value arises since the vehicle could be on either side of the wall. The measurement z_i is the true perpendicular distance r_i corrupted by zero-mean Gaussian noise w with variance w_r:

$$z_i = r_i + w, \quad w \sim N(0, w_r)$$

Suppose that at time t_k the robot is predicted to be in the location $\hat{\mathbf{x}}(k \mid k-1)$. The predicted location is used to suggest n walls to obtain range readings from. The sensor is directed to look for each of these walls, and the returned values are checked with an appropriate validation gate to yield validated range readings z_i for m of the n visible walls. We combine the validated range measurements to form a composite measurement vector $\mathbf{z}(k) = [x_1, \ldots, x_m]^T$. To relate this measurement vector $\mathbf{z}(k)$ to the vehicle position $\mathbf{x}(k)$, we define the non-linear function $\mathbf{h}$

$$\mathbf{h}[k, \mathbf{x}(k)] = \begin{bmatrix} |R_1 - x_k \cos\theta_1 - y_k \sin\theta_1| \\ \cdot \\ \cdot \\ \cdot \\ |R_m - x_k \cos\theta_m - y_k \sin\theta_m| \end{bmatrix} \tag{20}$$

Finally, a composite noise vector $\mathbf{w}(k)$ completes the measurement model:

$$\mathbf{z}(k) = \mathbf{h}[k, \mathbf{x}(k)] + \mathbf{w}(k), \quad \mathbf{w}(k) \sim N(0, \mathbf{R}(k)) \tag{21}$$

The function $\mathbf{h}[k, \mathbf{x}(k)]$ incorporates the prior information concerning the walls $\mathbf{p}_i$ for which validated range readings have been obtained at this time interval. So far we have ignored the data association problem, assuming that each range measurement comes from the appropriate wall. The viability of this assumption depends on how accurately the system model predicts the vehicle's movement; experimentally it has been justified so far.

Extended Kalman Filter Equations

The extended Kalman filter equations for the plant and measurement models described above in equations 17 and 21 use linearized plant and measurement models. The plant model $\mathbf{f}[k, \mathbf{x}(k)]$ is linearized about the current *estimated* state vector, $\hat{\mathbf{x}}(k \mid k)$. We accomplish this by defining the plant Jacobian, $\mathbf{f}_{\mathbf{x}}(k)$:

$$\mathbf{f}_{\mathbf{x}}(k) = [\nabla_{\mathbf{x}} \mathbf{f}^T(k, \mathbf{x})]^T_{\mathbf{x}=\hat{\mathbf{x}}(k|k)} = \begin{bmatrix} 1 & 0 & -T_k \sin\phi_k & cos\phi_k \\ 0 & 1 & T_k \cos\phi_k & sin\phi_k \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix}_{\mathbf{x}=\hat{\mathbf{x}}(k|k)} \tag{22}$$

The measurement model $\mathbf{h}[k, \mathbf{x}(k)]$ is linearized about the current *predicted* state vector, $\hat{\mathbf{x}}(k+1 \mid k)$. We define the measurement Jacobian $\mathbf{h}_{\mathbf{x}}(k+1)$ to be:

$$\mathbf{h}_{\mathbf{x}}(k+1) = [\nabla_{\mathbf{x}} \mathbf{h}^T(k, \mathbf{x})]^T_{\mathbf{x}=\hat{\mathbf{x}}(k+1|k)} = \begin{bmatrix} \pm\cos\theta_1 & \pm\sin\theta_1 & 0 & 0 \\ \cdot & \cdot & \cdot & \cdot \\ \cdot & \cdot & \cdot & \cdot \\ \cdot & \cdot & \cdot & \cdot \\ \pm\cos\theta_m & \pm\sin\theta_m & 0 & 0 \end{bmatrix} \tag{23}$$

The ± signs arise because $\mathbf{h}[k, \mathbf{x}(k)]$ contains the absolute value function fo: reasons given above. In calculating $\mathbf{f_x}(k)$, the appropriate sign for each wall is chosen based on the dead-reckoning estimate of the robot's position. The absolute value fu..ctions would make the Jacobian of $\mathbf{h}[k, \mathbf{x}(k)]$ unstable at sensing locations very close to the wall. This is not a problem in practice because the sensor is located at the center of the vehicle. The definitions of $\mathbf{f_x}(k)$, $\mathbf{h_x}(k+1)$, and $\mathbf{S}(k+1)$ are used in the extended Kalman filter equations (11) - (15) in a multi-target control structure.

Results

We avoid some of the filter initialization difficulties by starting the vehicle from an exactly-known initial location and setting the initial state covariance matrix $\mathbf{P}(0 \mid 0)$ to zero. Because the motion of the robot vehicle has a significant unpredictable component, the state covariance matrix $\mathbf{P}(k \mid k)$ grows considerably with time if no updates of position from the sensor are provided.

Three runs of the simulated system are shown in Fig. 9. The error ellipse is defined by the first four elements of $\mathbf{P}(k \mid k)$, and represents the uncertain·y in the prediction of the vehicle position (x_k, y_k). In Fig. 9(a), the speed noise v_T causes the uncertainty ellipse to grow with time in the direction of motion of the vehicle. The heading angle noise v_ϕ causes the uncert: inty ellipse to grow in the direction perpendicular to the vehicle's motion. *Our task for the measui ment pr·cess* is to decrease the size of this ellipse, and hence increase the confidence in the state estimate $\hat{\mathbf{x}}(k \mid k)$. Fig. 9(b) and (c) show the effects of measurements from one and two walls, respectively.

Future Woı k

In this implementation, using a single Kalman filter to derive the position of the robot is effective because

1. The environment has been known exactly *a priori*
2. The environment has no moving objects and no clutter.

In a practica! situation, an ultrasonic navigation system may need to deal with an unknown environment that contains unmodeled walls, unknown moving objects, and clutter. For this reason, in addition to the central Kalman filter for vehicle position, a multi-target framework that initializes and maintains tracks for individual targets in the environment is necessary. We plan to implement the three-layer strategy of Active Sensor Control as a mechanism for managing the multi-target tracking approach to navigation. In the sensor control framework, we shall develop a set of local attention processes, each tracking a particular target or searching for new targets, and carrying out some fast, local processing to facilitate this local control. The looking layer will perform the multi-target tracking *per se*, assigning local tracking resources to follow individual pre-identified targets and responding to new targets and unexplained events.

A DECENTRALIZED FILTER

FULLY DECENTRALIZED DECISION MAKING

Previous attempts at decentralized Kalman filtering ı.ave either assumed a completely centralized architecture (Fig. 1) in which all the sensors' observations are passed back to a central processing facility that fuses the data, [BF88], [DW76], [Cho79], or assumed *some* level of local embedded processing capability in each sensing node, but still however relying on a central processing facility to perform the global data fusion (ie. hierarchical decentralization) [HRL88], [Hen37]. Since both of these methods require a single central facility to perform overall data fusion they suffer from the associated problems: potential computational bottlenecks and the susceptibility to total system failure if the central facility should fail. Harris and White [HW87]

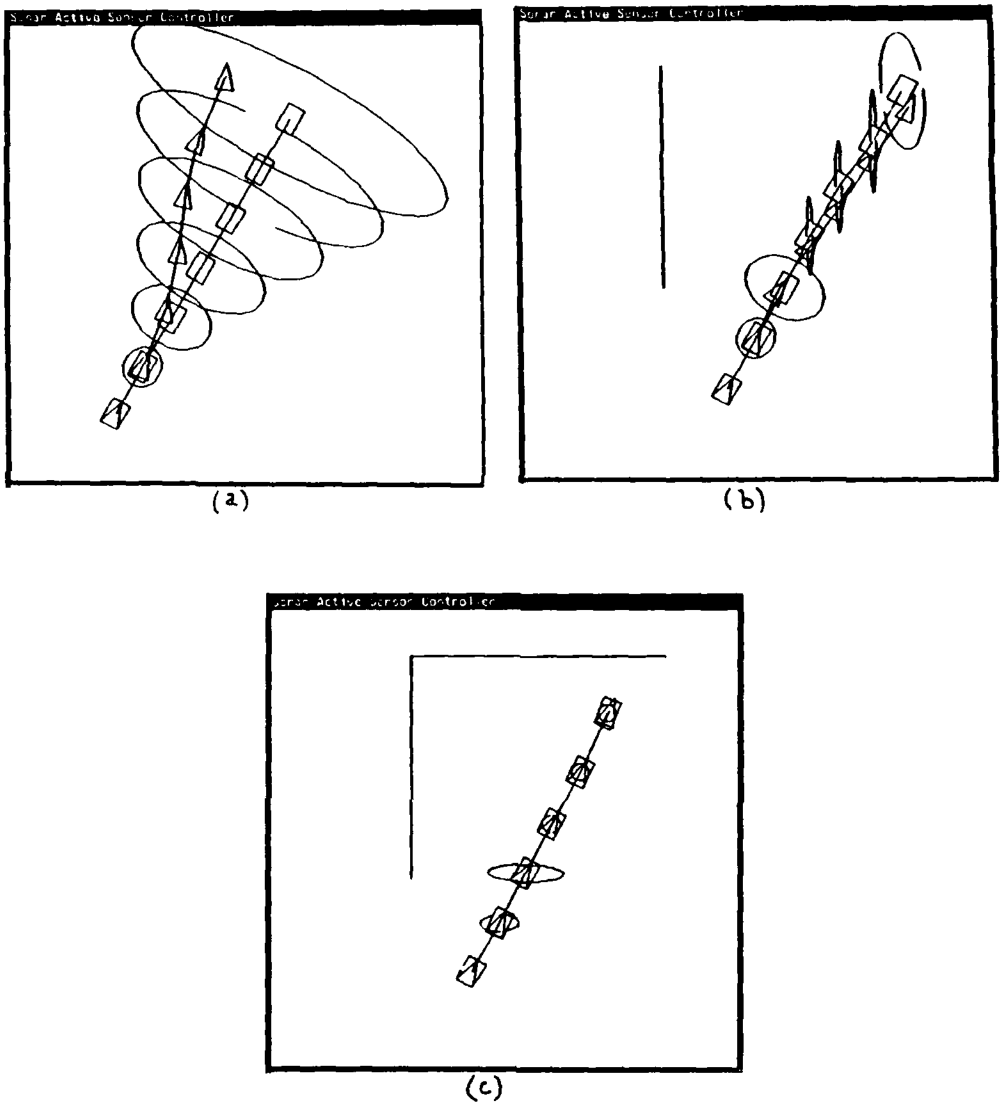

Sonar Active Sensor Controller
(a)
Sonar Active Sensor Controller
(b)
Sonar Active Sensor Controller
(c)

Figure 9: (a) A run with no validated measurements. The triangle represents the actual vehicle position and orientation (x_k, y_k, ϕ_k), the rectangle represents the estimated vehicle position and orientation, and the ellipse represents the uncertainty in the estimates of x_k and y_k. (b) A run taking range measurements from a single wall. The error ellipse shrinks perpendicular to the wall as the *a posteriori* confidence in the estimate of x_k and y_k increases with measurements. The wall comes into view and then disappears from view during the run, causing large effects in the error ellipse. (c) Measurements from two walls. Information from two directions reduces error.

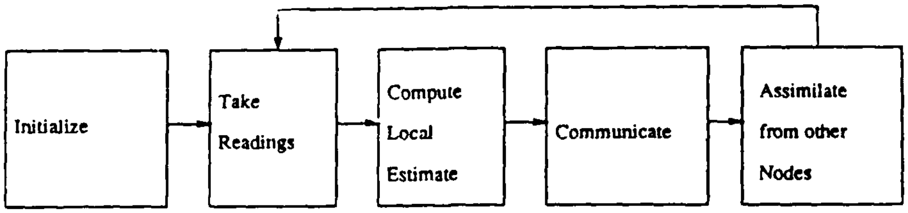


Figure 10: Algorithm for individual nodes.

give descriptions of algorithms for decentralized system architectures using blackboards for communication between nodes, but they suffer from the well-known problems associated with blackboard systems (such as keeping the blackboard current and free of redundant information).

Fully decentralized decision making [HD88] is advantageous to hierarchically decentralized and other forms of distributed system architectures because:

1. It allows complete parallelization of any algorithm. Therefore the system would be ideal for implementation on a parallel processing network (such as a transputer array).
2. It leads to a speed increase over other architectures by removing potential computational bottlenecks.
3. It gives a system that is very resilient to loss of one or more of its nodes (ie. a highly *survivable* system).

Distributed sensing and algorithms for sensor fusion that can be distributed amongst several sensing nodes have interesting applications in robotics, process plants and C^3I fields. The algorithm we present here is fully decentralized, requiring no central processing facility to perform data fusion [RD89]. The information communicated between nodes is simple and the equations for assimilation of other nodes' data are no more complex than those for a conventional Kalman Filter. Our filter reaches the same global optimum as a conventional Kalman Filter, but since this filter may be run simultaneously at the m sensing nodes of an m node system it performs faster.

The Decentralizing Algorithm

Fig. 10 shows the operations performed by each one of the nodes in the system. Each node is initialized with estimates of the state of the system together with an associated variance of that estimate. These initial values for state estimates and variances may be obtained in several ways and the methods used by the implementations given in this paper are outlined in later sections. After initialization, the main loop begins with each node taking a reading from its sensor of the state of the system. With this reading (and its associated variance) the node is able to compute conventional Kalman Filter equations to reach its own, new estimate of the state. Each node then broadcasts this estimate to the other nodes and receives information being broadcast to it. Last, each node computes assimilation equations to take into account the data it has just received; in fact, each node thus locally computes a global estimate.

This paper includes details of a linear version of the algorithm in which each node shares the same coordinate frame and has the same representation of the state of the system as other nodes. In the nonlinear case, each node, although measuring the same state, is in its own coordinate frame and uses its own representation of the state which may or may not coincide with the representation used by other nodes. This case is treated in full in [RD89] and is not derived here, although experimental results are given.

The Linear Algorithm

Consider the linear Kalman filter equations 4 - 7. To decentralize them we must make the following **Assumption** (from Hashemipour [HRL88]): Assume we can partition the observation vector into m subvectors of dimension m_i. Assume also that we can partion the **H** matrix conformably. (This means a node cannot make a full-rank observation of the system state.) We can therefore say that

$$\mathbf{v}_i(k) = [\mathbf{v}_i^T(k), \ldots \mathbf{v}_m^T(k)]^T$$

and that

$$E\{\mathbf{v}(k)\mathbf{v}^T(k)\} = blockdiag\{\mathbf{R}_i(k), \ldots, \mathbf{R}_m(k)\}$$
$$\mathbf{H}(k) = \{\mathbf{H}_i^T(k), \ldots, \mathbf{H}_m^T(k)\}$$
$$\mathbf{z}(k) = \{\mathbf{z}_i^T(k), \ldots, \mathbf{z}_m^T(k)\}$$

From this assumption we can also partition the **F** and **G** matrices and also the state estimates and variances, $\hat{\mathbf{x}}$ and **P**. This allows each node to implement its own local Kalman Filter for its own local estimate of the state.

Each node, i has a system model and takes observations that are individualized versions of eq. (1) - (9). That is, simply subscript the following variables in those equations with i: **x**, **z**, **F**, $\bar{\mathbf{F}}$, **G**, **H**, **Q**, **S**, **R**, **P**, **K**.

Derivation of Assimilation Equations

We now derive the equations each node computes to assimilate the information supplied by other nodes. References to equations (1) - (9) mean their local versions. From the **Assumption** above,

$$\mathbf{H}^T\mathbf{R}^{-1}(k)\mathbf{H} = \sum_{i=1}^{m}[\mathbf{H}_i^T(\mathbf{R}_i^{-1}(k)\mathbf{H}_i] \tag{24}$$

and also

$$\mathbf{K}(k)\mathbf{z}(k) = \mathbf{P}(k \mid k)\sum_{i=1}^{m}[\mathbf{H}_i^T\mathbf{R}_i^{-1}(k)\mathbf{z}_i(k)]. \tag{25}$$

From (7) we can say

$$\mathbf{H}_i^T\mathbf{R}_i^{-1}\mathbf{H}_i = \mathbf{P}_i^{-1}(k+1 \mid k+1) - \mathbf{P}_i^{-1}(k+1 \mid k). \tag{26}$$

Hence from (7), (24) and (26) we can write

$$\mathbf{P}^{-1}(k+1 \mid k+1) = \mathbf{P}^{-1}(k+1 \mid k) + \sum_{j=1}^{m}[\mathbf{P}_j^{-1}(k+1 \mid k+1) - \mathbf{P}_j^{-1}(k+1 \mid k)],$$

and by placing this assimilation equation at each node (decentralizing) we arrive at

$$\mathbf{P}_i^{-1}(k+1 \mid k+1) = \mathbf{P}_i^{-1}(k+1 \mid k) + \sum_{j=1}^{m}[\mathbf{P}_j^{-1}(k+1 \mid k+1) - \mathbf{P}_j^{-1}(k+1 \mid k)].$$

The summation over j does not include the term when $j = i$ since this has already been accounted for in (7).

Now premultiplying (26) by $\mathbf{P}_i(k+1 \mid k+1)$ and rearranging gives

$$\mathbf{I} - \mathbf{K}_i\mathbf{H}_i = \mathbf{P}_i(k+1 \mid k+1)\mathbf{P}_i^{-1}(k+1 \mid k). \tag{27}$$

Premultiplying (6) by $\mathbf{P}_i^{-1}(k+1 \mid k+1)$, using (27) and rearranging gives

$$\mathbf{H}_i^T \mathbf{R}_i^{-1} \mathbf{z}_i = \mathbf{P}_i^{-1}(k+1 \mid k+1)\mathbf{x}_i(k+\mathrm{i} \mid k+1) - \mathbf{P}_i^{-1}(k+1 \mid k)\mathbf{x}_i(k+1 \mid k). \tag{28}$$

Using (6), (27) and (28) and decentralizing gives

$$\begin{aligned}\hat{\mathbf{x}}_i(k+1 \mid k+1) = {} & \mathbf{P}_i(k+1 \mid k+1)[\mathbf{P}_i^{-1}(k+1 \mid k)\hat{\mathbf{x}}_i(k+1 \mid k) \\ & + \sum_{j=1}^{m} \{\mathbf{P}_j^{-1}(k+1 \mid k+1)\hat{\mathbf{x}}_j(k+1 \mid k+1) - \mathbf{P}_j^{-1}(k+1 \mid k)\hat{\mathbf{x}}_j(k+1 \mid k)\}\,].\end{aligned}$$

From (4) we can write

$$\mathbf{G}_i \mathbf{S}_i \mathbf{R}_i^{-1} \mathbf{z}_i = \mathbf{x}_i(k+1 \mid k) - \bar{\mathbf{F}}_i \mathbf{x}_i(k+1 \mid k+1) \tag{29}$$

and noting that from the **Assumption** we can write

$$\mathbf{G}\mathbf{S}\mathbf{R}^{-1}\mathbf{z} = \sum_{i=1}^{m} [\mathbf{G}_i \mathbf{S}_i \mathbf{R}_i^{-1} \mathbf{z}_i]$$

we can derive

$$\hat{\mathbf{x}}_i(k+1 \mid k) = \bar{\mathbf{F}}_i(k)\hat{\mathbf{x}}_i(k \mid k) + \sum_{j=1}^{m} \{\hat{\mathbf{x}}_j(k+1 \mid k) - \bar{\mathbf{F}}_j(k)\hat{\mathbf{x}}_j(k+1 \mid k+1)\}.$$

Assimilation Equations

To reiterate: each node takes readings from its sensor, computes its local version of equations 5 - 6, communicates to all o'her nodes (see below) and then computes the equations given below.

Prediction:

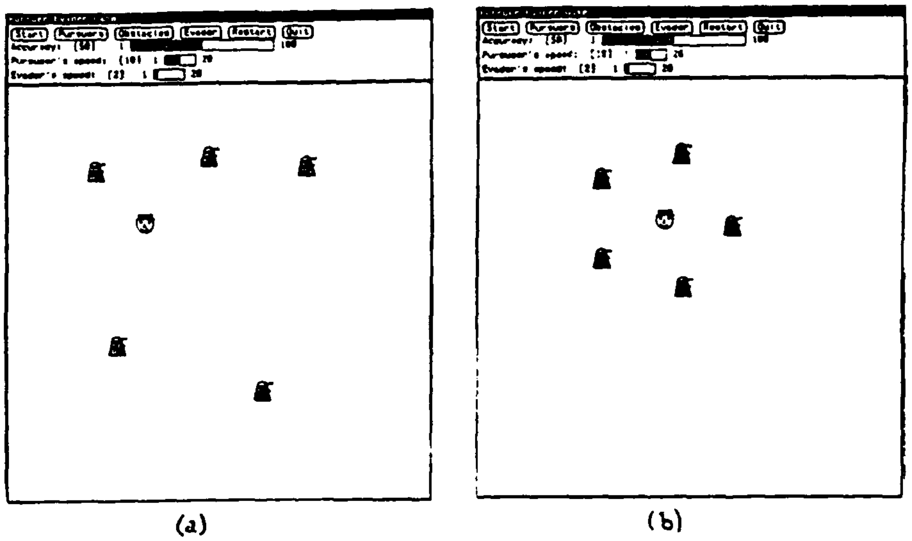
$$\hat{\mathbf{x}}_i(k+1 \mid k) = \bar{\mathbf{F}}_i(k)\hat{\mathbf{x}}_i(k \mid k) + \sum_{j=1}^{m} \{\hat{\mathbf{x}}_j(k+1 \mid k) - \bar{\mathbf{F}}_j(k)\hat{\mathbf{x}}_j(k+1 \mid k+1)\} \tag{30}$$

$$\mathbf{P}_i(k+1 \mid k) = \bar{\mathbf{F}}_i(k)\mathbf{P}_i(k+1 \mid k+1)\bar{\mathbf{F}}_i^T(k) + \mathbf{G}_i(k)\bar{\mathbf{Q}}_i(k)\mathbf{G}_i^T(k) \tag{31}$$

Update:

$$\begin{aligned}\hat{\mathbf{x}}_i(k+1 \mid k+1) = {} & \mathbf{P}_i(k+1 \mid k+1)[\mathbf{P}_i^{-1}(k+1 \mid k)\hat{\mathbf{x}}_i(k+1 \mid k) \\ & + \sum_{j=1}^{m} \underbrace{\{\mathbf{P}_j^{-1}(k+1 \mid k+1)\hat{\mathbf{x}}_j(k+1 \mid k+1) - \mathbf{P}_j^{-1}(k+1 \mid k)\hat{\mathbf{x}}_j(k+1 \mid k)\}}_{\textit{state error info}}\,]\end{aligned} \tag{32}$$

$$\mathbf{P}_i^{-1}(k+1 \mid k+1) = \mathbf{P}_i^{-1}(k+1 \mid k) + \sum_{j=1}^{m} \underbrace{\{\mathbf{P}_j^{-1}(k+1 \mid k+1) - \mathbf{P}_j^{-1}(k+1 \mid k)\}}_{\textit{variance error info}} \tag{33}$$



Accuracy: [50] 1 100
Pursuer's speed: [10] 1 20
Evader's speed: [2] 1 20

Figure 11: (a) A typical starting configuration of pursuers and the evader. (b) The pursuers have surrounded the evader.

using $\hat{\mathbf{x}}(0 \mid -1) = \hat{\mathbf{x}}_0$ and $\mathbf{P}(0 \mid -1) = \mathbf{P}_0$ for initialization of each node.
(The summations above are over every node j from $j = 1$ to $j = m$ except for when $j = i$.)
The terms *state error info* and *variance error info* are the values transmitted by each node to each other node during the communication step of Fig. 10. The information to be communicated is not complex (one vector and one matrix per iteration), and the data fusion equations are no more complex than the local estimate update equations.

IMPLEMENTATION

The algorithms have been implemented on a SUN 3.0 computer simulating several nodes running simultaneously. The application is a pursuer-evader game in which there are a number of independent pursuers attempting to surround and trap a moving evader. The pursuers and evader move on a two dimensional surface: the evader only moves horizontally and with a constant velocity. Each pursuer is an independent mobile sensing node taking noisy observations of the single moving evader. Each pursuer has its own model of the system (ie. the evader's motion pattern) and they all communicate between themselves to arrive at a global decision of the location of the evader so that they can then decide where best to move to surround it (see Fig. 11).

Linear Case

In this case each pursuer takès full-state $\mathbf{z} = (x, y, \dot{x})^T$ observations of the evader, corrupted by noise (the noisiness of a node's readings being proportional to the distance between that pursuer and the evader) and each has a correct model of the evader's dynamics. The evader is travelling in the horizontal direction at constant velocity. After taking readings the pursuers evaluate local versions of equations 5 - 6 and then calculate the data they must transmit all the other pursuers. Each communicates this data to each other pursuer and collect others' *variance error info* and *state error info* to arrive at a global estimate for the (x, y) position of the evader and also its velocity in the x direction. The Kalman filter is initialized by setting the observed velocity of the evader to zero for the first five iterations to allow the filter to settle before the noisy observations of the velocity are taken into account.

Results

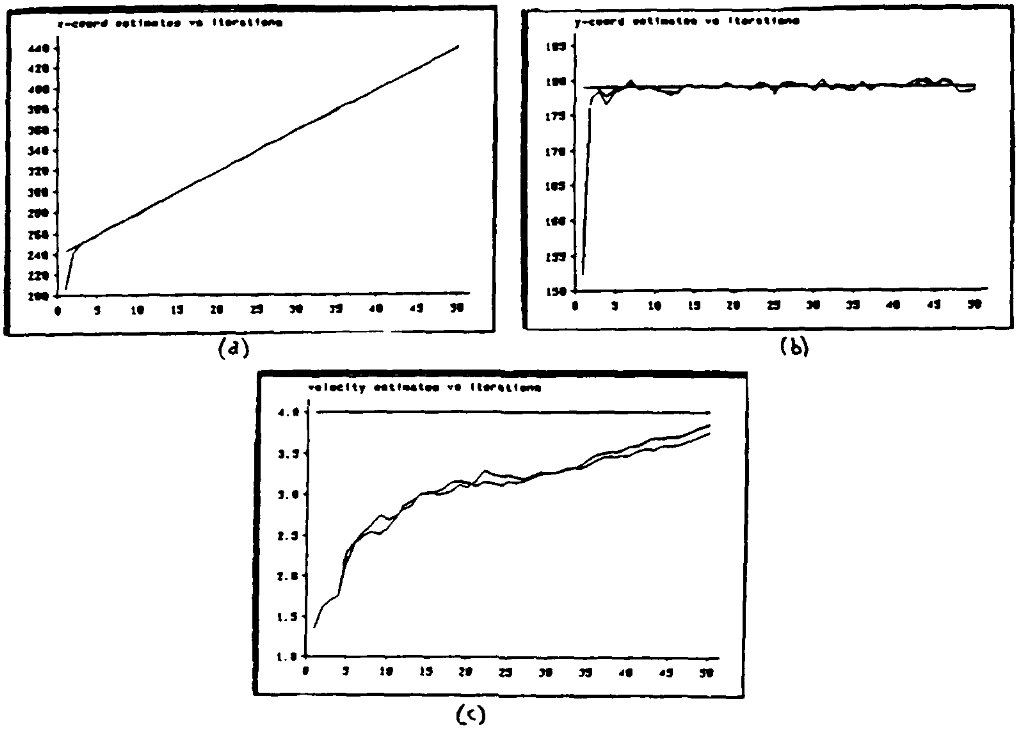


Figure 12: (a) Linear case x-coordinate estimate (actual = solid rising line). (b) Linear case y-coordinate estimate (actual = 178). (c) Linear case horizontal velocity estimate (actual = 4.0).

Figs. 12 and 13 are graphs of the global estimates from two pursuers (only two are shown for clarity) and the evader's actual parameters plotted against number of iterations. These results show:

1. Each node arrives at the same (albeit initially inaccurate) global estimate from the very first iteration. Therefore, by communication the evaders have reached a common group consensus of the position of the evader.
2. The global estimates for the (x, y) positions soon converge to very close to the actual values and continue to track it well. The filter converges faster than a conventional Kalman Filter since it is able to process m lots of sensors' information simultaneously.
3. The global estimate for the velocity of the evader $\dot{\mathbf{x}}$ converges more slowly to the actual value but this is to be expected since $\dot{x}$ is small compared to the observation noise and also differentiation always amplifies noise.

Nonlinear Case

In this case each pursuer has an $(r, \theta, \dot{r}, \dot{\theta})$ state representation of the position of the evader, which is moving in the horizontal direction with a constant velocity. However, in order to simulate a realistic case where each pursuer is a photodiode based device, *angle only measurements are taken*. No range data is available from the sensors so communication between the pursuer is essential for pursuers to arrive at a full state vector. Each pursuer takes its sensor measurement and constructs its own view of the state of the object. Each pursuer knows where it is and where all the other pursuers were from the last communication step so each pursuer is able to transmit appropriately transformed variance error and state error information to the others. (Also each pursuer transmits its current position to all the others). The assimilation equations

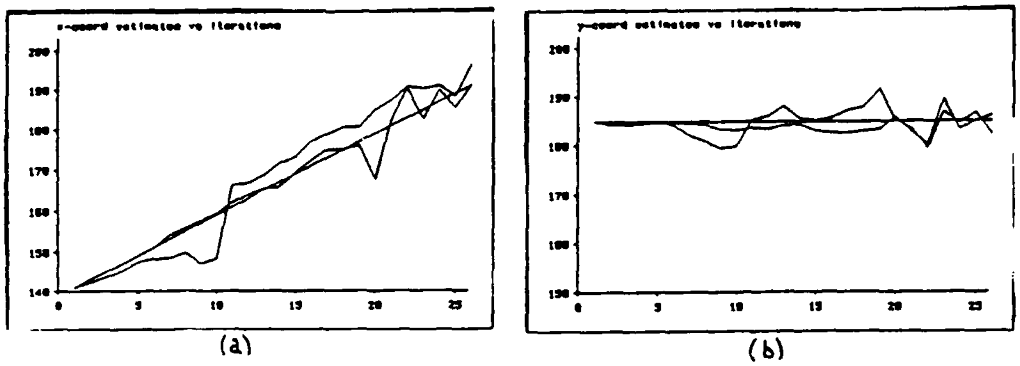


Figure 13: (a) Non-linear case x-coordinate estimate (actual = solid rising line). (b) Non-linear case y-coordinate estimate (actual = 185).

are computed and each pursuer ends up with a full state representation of the evader. This 'filling in' of the empty states occurs because if one computes the linear weighted sum of a node's state estimate $\mathbf{x}_i$ and another node's transformed estimate ${}^i\mathbf{x}_j$ the new state vector $\mathbf{x}_i$ obtained is full rank (ie. the intersection of the angle estimates of the two nodes has been solved to give range data) [Dur88]. If

$$\mathbf{x}_i = ({}_i\Sigma_x \mathbf{x}_i + {}^j_i\Sigma_x {}^i\mathbf{x}_j)({}_i\Sigma_x + {}^j_i\Sigma_x)^{-1}$$

where $\mathbf{x}_i$ and $\mathbf{x}_j$ are (r, θ) state vectors with initially no values for r and ${}_i\Sigma_x$ and ${}_j\Sigma_x$ are the associated information matrices with zeroes in the range information positions one obtains the correct estimate for r in $\mathbf{x}_i$. The filter required five steps to arrive at reasonably accurate estimates for the r and $\dot{r}$ terms for each pursuer. Then it was able to follow the evader, updating its estimates for $\dot{r}$ and $\dot{\theta}$ on each iteration.

Results

Results are shown in Fig. 13, which shows the (x, y) position of the evader and the (x, y) estimates of two pursuers. Only two pursuers' estimates are shown for clarity. The (r, θ) state representations of each pursuer have been converted to cartesian form for these plots, simply so that the estimates of two pursuers may be directly compared. The results show that:

1. The pursuers are able to track the evader despite each pursuer taking only a partial estimate of the state. By communication between themselves they are able to arrive at a full state vector estimate for the position of the evader.
2. The pursuers are in agreement over the position of the evader. The non-linearities in the system (ie. using an (r, θ) representation for a linear cartesian motion) cause the slight error between the two pursuers' estimates.

The Future

The algorithm is currently being implemented in OCCAM on an array of sensors, each with a photodiode based angle detector and its own transputer. This will enable the algorithm to be tested on real data and allow experiments with sensors returning measurements asynchronously. Also the algorithm may be extended to allow tracking of several objects simultaneously.

The motivation behind this work was to attempt to discover and set down mathematically what pieces of information must be communicated between a group of sensors in order for them to reach a conclusion. This problem must now be investigated further and beyond the limits of just a simple linear tracking filter algorithm. It appears that the multi-sensor network we have analysed here may be thought of as a complex

Markov chain with information propagating through each node. This line of research borders onto the wider issue of sensor models and may throw some light on how best to model complex sensors such as CCD cameras in order that the value of the information returned by a sensor in any arbitrary situation may be assessed by other sensors. Being able to model sensors in this way would lead to the possibility of building intelligent sensing nodes that could be connected together as a fully decentralized network.

DISCUSSION

OPTIMAL ESTIMATION AND ACTIVE INTELLIGENCE

Optimal estimation techniques have at least three distinct roles to play in real-time sensorimotor systems.

1. They can be used as the basic paradigm for estimating the state of systems *internal* to the observer. Estimating external states is akin to *perception*.
2. They can be used to estimate the state of systems *internal* to the observer. Estimating internal state involves aspects of *proprioception* (using information from internal sensors), but can also involve sensing the outside world, especially to determine dynamic observer parameters such as location and velocity.
3. They can be used as low-level utilities in service of several aspects of perception or action.

Control Styles in Perception

Top-down (expectation-driven, hypothesis-verification) methods cope with the inherent underdetermined and computationally intensive nature of perception by using apriori knowledge to constrain the space of interpretations for perceptual data. In a navigation context, these methods correspond to map-guided route-finding, perhaps landmark recognition and similar tasks in which an internal model exists and the input is expected.

One important role of perception is to cope with the unexpected. This seeming truism is often ignored, and has deep implications for computational perceptual models. In particular it implies that tops-down control strategies are by themselves inadequate. In a navigational context, obstacle avoidance illustrates this role. The complementary control strategy is *bottom-up*, or data-driven: Here the style is often a fixed order of processing of input data (say by successive levels of feature detection and extraction) leading to increasingly abstract levels of description of the input. As technology improves it is becoming possible to achieve the massive data-processing effort in real time, and the practical considerations that have partially motivated the tops-down approach are vanishing (see, e.g. [Bro88]).

In one sense, the Kalman filter is an example of expectation-driven perception. By definition it incorporates explicit models of dynamics and noise. The strength of the Kalman filter for estimation is that it has these models at its disposal, but requiring them limits the sorts of perceptual jobs that the Kalman filter can reasonably be expected to perform. The severe tops-down requirements can be mitigated to some extent, and at some cost, by such measures as running several different filters embodying different assumptions in parallel, switching between filters when lack of fit motivates such a switch, allowing the filter to estimate control inputs to the plant, etc., as we have seen in earlier sections.

Despite such seemingly sophisticated adaptive capabilities, the extensive literature on Kalman filtering applications (e.g. [BF88,Rei79,Abu86,Bar78,Gel73,Hal84,Ken81,Mor77,MCTW86,Ed83] reveals that the *perceptual* tasks most often attempted are those in which the plant (often *target*) follows well-known and rather simple (e.g. ballistic) dynamic laws, and in which the target is modeled as a point in space. The typical perceptual task is tracking the (perhaps maneuvering) target (perhaps in clutter). Thus the perceptual task (*data association*, or *segmentation*), consists of the twofold problem of linking measurements together into tracks and ignoring spurious data. The basic Kalman filter mechanism provides help in the way of quantified measures of uncertainty, surprise, information, expectation, etc. but provides nothing directly to cope with the familiar problems of controlling search in interpretation space. The track-splitting, NNSF, PDAF, etc. approaches all have analogs in the edge-linking problem in computer vision, for example.

Dissatisfaction with the paradigm of expectation-driven low-level perception, combined with skepticism about the efficacy of local bottom-up segmentation, has motivated other algorithms for target tracking. Their goal is often to accomplish perceptual grouping using global, expectation-driven metrics of grouping quality, as well as to cope with input in a more data-driven way [Kuc87]. However, much remains to be done here.

Perception: Estimating External State

It seems that if optimal estimation techniques are to be a paradigm for *perception*, at least the following conditions must apply.

1. The dynamics of the objects must obey known, predictable laws.
2. The noise mean and covariance properties of the domain dynamics and of the measurement system must be known apriori.
3. The data may arise from several information sources — the Kalman filtering technique provides a principled way to combine (fuse) them.
4. The raw perceptual input must be processed to yield a measurement vector containing information about the state of the observed objects. If objects are more complicated than single points, this step may call for solving "the vision problem" in order to do tracking. An extreme example is reliable tracking of a face in a crowd, using data from a face-recognizer. The point is that even such basic vision tasks as region-finding are not well understood, and should not be lightly suggested as "preprocessing" for a tracker.
5. Measurement data must be available over a significant interval, probably tens of time-steps for reasonably complex domain dynamics.
6. Dealing with complex perceptual events in real time will call for substantial computational resources.

Proprioception: Estimating Internal State

The other main use for Kalman filters is for internal state estimation, in aid of complex, often adaptive, control (e.g., [Ed83]). Such *proprioception* is not divorced from perception: A vehicle can determine its position from fixed beacons (say landmarks or stars) by tracking them and interpreting the data with a Kalman filter. In practice, the contrast between the sophistication of the dynamic models for plants that are to be estimated and controlled and the models of external targets is dramatic (for a plant to be controlled, sometimes 80 state variables, for a target to be tracked, perhaps six). Presumably the observer's state description equations are relatively stable, and devoting computational and analytic resources to precise observer description is a good long-term investment that will pay off in better information about and control over its state. For another thing, the observables themselves are under more control. A roving vehicle can observe bar-coded reflectors as location beacons, or can track static features such as walls or furniture, whose apparent motion arises more or less predictably from observer motion.

Thus the proprioception problem is inherently more "expectation driven" than the perception problem, and Kalman filtering techniques may well be an appropriate paradigm since the following conditions apply.

1. The dynamics of the observer obeys known, predictable laws. They may be complex to model but there is only a single, known system to characterise, and it makes sense that the observer is something the observer itself knows best.

2. Proprioception may be provided by on-board sensors such as tachometers, odometers, shaft encoders, etc.

3. The noise mean and covariance properties of the observer dynamics and its measurement system can be experimentally determined "off-line", as can the properties of the expected visual stimuli.

4. The raw perceptual input must be processed to yield a measurement vector containing information about the state of the observer, but the observer can choose to interpret a limited set of stimuli by customized methods if internal state estimation is the only goal.

5. Presumably measurement data is available for a significant interval, on the order of the "lifetime" of the observer, rather than of the lifetime of the unexpected visual phenomena that occur in perception.

6. Computational requirements may not be as severe since the update rate needed for proprioception may be lower than that for perception.

THE FUTURE

We plan to use both aspects of estimation in work at Rochester to integrate real-time vision and motion with high-level planning in a hierarchical parallel system. An early step will be the inclusion of estimation techniques in the robot's gaze control system [Bro89].

Acknowledgements

C. Brown's stay in Oxford is partially funded by the DARPA U.S. Army Engineering Topographic Laboratories Grant DACA76-85-C-0001 and the Air Force Systems Command (RADC, Griffiss AFB, NY) and Air Force OSR Contract F30602-85-C-0008, which supports the Northeast Artificial Intelligence Consortium. J. Leonard is supported by a Thouron fellowship. The assistance of Prof. J. Michael Brady and the RRG research and administrative staff is gratefully acknowledged.

References

[Abu86] Ahmed S. Abutaleb. Target image tracking using a nonlinear filter based on pontryagin minimum principle. *IEEE Transactions on Automatic Control*, AC-31(12):1170–1173, December 1986.

[Bar78] Y. Bar-Shalom. Tracking methods in a multi-target environment. *IEEE Transactions on Automatic Control*, AC-23(4):618–626, August 1978.

[BF88] Y. Bar-Shalom and T.E. Fortmann. *Tracking and Data Association*. Academic Press, 1988.

[Bro88] C. M. Brown. *The Rochester Robot*. Technical Report 257, University of Rochester, September 1988.

[Bro89] C. M. Brown. Gaze controls with interactions and delays. In *DARPA IU Workshop*, 1989.

[Cho79] C.Y. Chong. Hierarchical estimation. In *2nd MIT/ONR CCC Workshop*, Monterey, CA., 1979.

[Dur88] H.F. Durrant-Whyte. Uncertain geometry in robotics. *IEEE J. Robotics and Automation*, 4(1):23-31, 1988.

[DW76] K.P.Dunn D. Willner, C.B.Chang. Kalman filter algorithm for a multi-sensor system. In *15th IEEE Conf. Decision Control*, Clearwater, Florida, 1976.

[Ed83] II.W. Sorenson - Guest Ed. Special issue on applications of kalman filtering. *IEEE Transactions on Automatic Control*, AC-28(3), March 1983.

[Gel73] Arthur C. Gelb. *Applied Optimal Estimation*. The MIT Press, 1973.

[IIal84] John C. T. Hallam. *Intelligent Automatic Interpretation of Active Marine Sonar*. PhD thesis, University of Edinburgh, 1984.

[IID88] G. Hager and H.F. Durrant-Whyte. Information and multi-sensor coordination. In *Uncertainty in Artificial Intelligence*, pages 381–394, North Holland, 1988.

[IIen87] T. Henderson. *Workshop on Multi-Sensor Integration*. Technical Report UUCS-87-006, U. Utah Computer Science, 1987.

[IIRL88] II.R. IIashemipour, S. Roy, and A.J. Laub. Decentralized structures for parallel kalman filtering. *IEEE Trans. Automatic Control*, 33(1):88–93, 1988.

[IIW87] C.J. IIarris and I. White. *Advances in Command, Control and Communication Sytems*. IEE Press, 1987.

[Ken81] Richard J. Kenefic. Optimum tracking of a maneuvering target in clutter. *IEEE Transactions on Automatic Control*, AC-26(3), June 1981.

[Kuc87] Robert M. Kuczewski. Neural network approaches to multi-target tracking. In *IEEE First International Conference of Neural Networks*, June 1987.

[MCTW86] Shozo Mori, C. Chong, E. Tse, and R. Wishner. Tracking and classifying multiple targets without a priori identification. *IEEE Transactions on Automatic Control*, AC-31(5), May 1986.

[Mor77] C. L. Morefield. Application of 0-1 integer programming to multitarget tracking problems. *IEEE Transactions on Automatic Control*, AC-22(6), June 1977.

[RD89] B. Rao and II. Durrant-Whyte. A fully decentralized algorithm for multi-sensor kalman filtering. In *28th IEEE Conf. on Decision and Control*, December 1989.

[Rei79] Donald B. Reid. An algorithm for tracking multple targets. *IEEE Transactions on Automatic Control*, AC-24(6), December 1979.

Objective Functions for Feature Discrimination: Applications to Semiautomated and Automated Feature Extraction *

Pascal Fua   Andrew J. Hanson

Artificial Intelligence Center
SRI International
333 Ravenswood Ave.
Menlo Park, CA 94025

Abstract

In a companion paper, we have proposed a way of exploiting information-theoretic objective functions to evaluate the correspondence between a generic model language and shape hypotheses in a digital image. These objective functions balance a model's goodness of fit to the photometric evidence against its geometric quality. Here, we describe applications of the objective function approach to the extraction of features from aerial imagery using both operator-guided cueing and automated hypothesis-generation methods. We formulate generic models for buildings and present experimental results on a variety of difficult aerial images.

Introduction

The goal of a feature-extraction system is to find the best labeling of a scene in terms of a particular set of models. Given unlimited computing resources, we might proceed simply by evaluating an objective function such as that proposed in the companion paper [9] for every possible combination of pixels and keeping those with the best scores. Since such exhaustive search is not feasible in practice, interactive or intelligent automated procedures are required to generate likely feature hypotheses. An objective function can then be used to optimize the spatial characteristics of single hypotheses or to rank a collection of static feature candidates. In order to achieve reasonable rankings of an exhaustive set of hypotheses, the objective function would typically have to embody a very complex semantic model; when excellent initial hypotheses are available from human input or from an hypothesis generator, simpler models may be sufficient.

In this paper, we begin by demonstrating the application of the objective function approach in a semiautomated environment where the human operator cues the system by providing a rough sketch of the feature of interest. Next, we show how, using a relatively simple generic model for buildings in aerial imagery, similar cues can be automatically generated and optimized, resulting in a ranked set of feature labels.

Interactive or semiautomated systems are of special importanre because no automated hypothesis-generation system is likely to approach human performance in the near future; practical systems should provide the opportunity to exploit human context knowledge and intuition to focus the attention of automated optimization systems such as ours.

An effective approach for fully automated systems is to generate candidates that correspond to *local maxima* of the model score expressed by the objective function. Then the model plays a uniform and consistent role in both hypothesis generation and evaluation, and the objective function itself serves ultimately to choose the most likely global maxima of the score.

The objective function approach shows its strength relative to other approaches in two particular circumstances:

*This research was supported in part by the Defense Advanced Research Projects Agency under Contract Nos. MDA903-86-C-0084 and DACA76-85-C-0004.

- **Difficult Data.** When the photometric evidence in the images is noisy and characterized by occlusions, shadows, and ambiguous edges, standard edge operators and segmentation procedures will rarely generate the unambiguous evidence needed to parse the scene. Objective functions provide a means for ranking large numbers of ambiguous possible parses.
- **Generic Models.** When a straightforward template model is adequate to search for the desired features, simpler approaches like the Hough transform [1] may suffice. An objective function approach that includes *geometric quality* measures balanced with the photometric information is much more important when the geometry of the model is *generic*, rather than fixed.

Application Domain: Cultural Features. As an application domain, we examine the extraction of rectilinear, presumably cultural, features from aerial imagery. This seems to be the simplest nontrivial model that exercises all the components of the objective function approach, and therefore is an excellent laboratory for experiments. Much more complex models can in principle be formulated with no change to the basic framework; we have carried out examples of several other models, including roads, vegetation, and three-dimensional buildings with shadows [6,7]. More details of applications using other models will be the subject of later work.

Other work on the cultural feature problem includes that of Huertas and Nevatia [23], which is a strict edge-based approach, a number of investigators such as Ohta, et al., [24] who concentrate on segmentations and region-based methods, and McKeown and Denlinger [21], who merge road intensity profiles with edge-tracking to delineate roads in aerial imagery. Our work assimilates many basic ideas from such approaches, unifies them into a framework that balances the experimental image information against abstract model expectations, and provides a theoretically justifiable objective function that distinguishes among competing hypotheses.

We begin by reviewing our objective function approach and our photometric and geometric models. Next, we investigate the application of the method to operator-guided refinement of two-dimensional features with various geometric constraints. Then we describe the application of subsets of the objective function to the critical procedures needed by an automated hypothesis-generation system for the discovery of buildings in aerial imagery. Finally, we apply the approach to a set of challenging aerial images and evaluate the results.

Review of the Objective Function Formulation

A Model for Buildings

We model buildings in aerial imagery as rectilinear objects with a planar intensity distribution, possibly with anomalies, and distinct edges. If stereo information is available, the model requires the building roofs to be elevated above the background. The goodness of fit of the data to the model is measured by the information-theoretic length of the encoded description of the data in terms of the model; the shorter the encoding, the better the fit.

The choice of the planar area model for buildings is based on simplicity and an experimental observation. Theoretically, one would expect building roofs to be planes that were approximately Lambertian reflectors, yielding constant intensity patches in the image. Experimentally, the combination of photometry, film processing characteristics, and digitization artifacts that characterize the vast majority of the digital aerial images at our disposal do not produce constant intensity patches, but patches that are much better described by a plane in intensity space. The plane is also the closest generalization of the constant intensity model that is easily computed. Invoking theoretical corrections for non-Lambertian reflectors would be much less justifiable, since we have no information whatever about the distribution of roof materials in our images, and in fact cannot know for certain that the observed effect does not derive from some step in the processing or digitization processes, as opposed to non-Lambertian reflectance. Finally, while the constant model produces verifiably poor results directly traceable to the area term in the model, misinterpretations found when using the planar model appear to come from many sources, so that there is no phenomenological justification for going beyond the planar model at this time.

As noted in [9], the technique used to discount anomalies can be critical for the local maximality of the score. In low resolution imagery, the techniques described in [9] have proven adequate. However, in high resolution images, an explicit representation of anomalies such as shadows and occlusions enhances the performance of the objective function. In this paper, we model such structured anomalies by allowing model instances to have holes whose encoding cost is taken to be the cost of encoding their boundaries.

In effect, this means that an area with an internal blotch that is very expensive to encode should be compared with a model of the same area that has an identically shaped cutout; the cheapest of the two encodings is the preferred interpretation. This approach causes the objective function to prefer to treat random noise by encoding each pixel, while choosing to cut out large anomalous areas, encoding only their shapes; these outlined areas would be prime candidates for possible later treatment using models for shadowing and occluding structures.

Evidence Measures

We have proposed in [9] a class of objective functions based on the *minimal description length* principle of Rissanen [25,26] that balances the photometry-based likelihood of a particular scene-labeling against the elegance and appropriateness of the geometric shape models used to propose a given set of labels. This approach has much in common with the information-theory approaches to segmentation of Leclerc [19] and of Georgeff and Wallace [11], and the decision theory approach to region labeling of Feldman and Yakimovsky [3].

A specific instance of the building model described in the previous subsection is characterized by its area A and its boundary length L. We model the area by fitting its intensities to a plane, histogram the deviations from the plane, and represent these deviations as a single gaussian peak of variance σ such that n pixels lie within the peak and $\bar{n}$ are considered outliers. Similarly, the boundary L is partitioned into two components, l pixels that are maxima of the edge gradient and thus satisfy our definition of an edge pixel, and $\bar{l}$ that fail to be maxima. Thus $A = n + \bar{n}$ and $L = l + \bar{l}$. Stereographic information is modeled by projecting a candidate model instance in one image to the other image and including separate terms in the objective function for the resulting area A_2 and boundary L_2.

The total score for a set of non-conflicting model hypotheses is

$$S(m_1, m_2, \ldots, m_N) = \sum_i S(m_i), \tag{1}$$

where the score for each model instance is the difference between what we call the *effectiveness* F of the photometric model and the geometric cost G. The effectiveness may be understood as the number of bits saved by representing the data with or without the model, and thus measures the goodness of fit of the data to the model. Separating the area, edge, and stereo terms in the model score, we write

$$S(m_i) = F_{i,A} + F_{i,E} + F_{i,S} - \gamma G_i, \tag{2}$$

where

$$\begin{aligned} F_A &= (8 - k_A)\frac{A}{s^2} && (3)\\ F_E &= (1 - k_E)\frac{L}{s} && (4)\\ F_S &= F_{A_2} + F_{E_2} && (5)\\ G &= c + \frac{L}{s} && (6) \end{aligned}$$

are area, edge, stereo, and geometric complexity measures, respectively. k_A is the number of bits needed to encode the distribution of pixels differing from the chosen planar model,

$$k_A A = n(\log \sigma + c) + 8\bar{n} + E(n, \bar{n}), \tag{7}$$

k_E is the number of bits needed to specify whether or not a pixel is a maximum of the image gradient,

$$k_E = E(l, \bar{l}), \tag{8}$$

and G represents a model for the boundary geometry that penalizes excessive length in the boundary L.

Here σ is the standard deviation of the main gaussian peak, $\bar{n}$ is the number of anomalous pixels, and $E(a,b) = -a \log \frac{a}{a+b} - b \log \frac{b}{a+b}$ is the entropy of the partition of a set of pixels into two distinct sets.

The scale s serves to normalize the intrinsic "complexity content" of the evidence in the image; s may also be viewed simply as the inevitable heuristic parameter that converts the pixel-based measures A and L from length dimensions into dimensionless numbers that can be interpreted in terms of probabilities. The geometry coefficient γ governs the trade-off between the pixel-based evidence and the tendency to hallucinate the desired shape model.

Operator-Initiated Shape Refinement

Procedures that efficiently locate the nearest local optimum of the objective function are required for the following purposes:

- Testing the nature and effectiveness of particular models incorporated into the objective function.
- Improving the characteristics of automatically generated feature hypotheses.
- Relieving human operators of the burden of metrically accurate feature delineation by automatically optimizing a rough sketch.

We address this problem by describing object contours as geometrically constrained curves moving in a potential V that can deform themselves to local maxima of the objective function; the resultant outline will then conform to the nearest object in the image that corresponds to the model represented by the objective function. Such curves were originated by Terzopoulos, Kass, and Witkin as "snakes" [16,29]. In their approach, boundaries are described as polygonal curves with a score that includes geometrical constraints and a measure of edge strength. "Snakes" do not take into account any photometric evidence outside the edge; they yield good results only if the initial position of the curve is close enough to the boundary of the object to be influenced by its edges. Since we also use *interior area information*, our curves can easily grow or shrink if the initial position is very inaccurate. By integrating more information and incorporating anomaly discounting, our algorithm also becomes more stable and less sensitive to photometric anomalies.

The optimization procedure for our objective function is extremely time-consuming on serial machines, but has been implemented in real time on the Connection Machine$^{TM 1}$ [8].

The Potential

In theory, the potential used by the optimization procedure should be computed using the objective function. In practice, however, the objective function used for scoring is inappropriate for snake-like optimization procedures because neither the edge measure nor the geometry measure are smooth enough to form a potential that acts over a reasonable distance. Therefore, for the purpose of optimization, the geometry is imposed as an external constraint, while the edge effectiveness is replaced by

$$F_{\text{grad}} = +\frac{1}{s} \sum_{\text{curve}} \log \begin{cases} \frac{g(x,y)}{g_0} & \text{if } g > g_0 \\ 0 & \text{otherwise,} \end{cases} \tag{9}$$

where

$$g(x,y) = \left(\frac{\partial I}{\partial x}\right)^2 + \left(\frac{\partial I}{\partial y}\right)^2 \tag{10}$$

It can be shown [10] that points on curves that maximize F_{grad} are local maxima of the edge gradient and therefore also maximize F_E.

Deformable Models in Two Dimensions

To find local maxima of the potential V, where $V = F_A + F_{\text{grad}}$, we describe object contours as deformable closed curves defined by an ordered list of contiguous points C represented by the vector X of their integer x coordinates and the vector Y of their y coordinates. During each iteration of the optimization procedure described below, X and Y are updated. C is then recomputed by drawing lines between points that are no longer contiguous and merging points that have identical coordinates, thereby generating new vectors X and Y. The edge effectiveness F_E is computed using those new boundary pixels and the area effectiveness F_A of the pixels enclosed by the boundary but not belonging to it. In this way the contour can shrink or expand as required to optimize the objective function.

[1] Trademark, Thinking Machines, Inc.

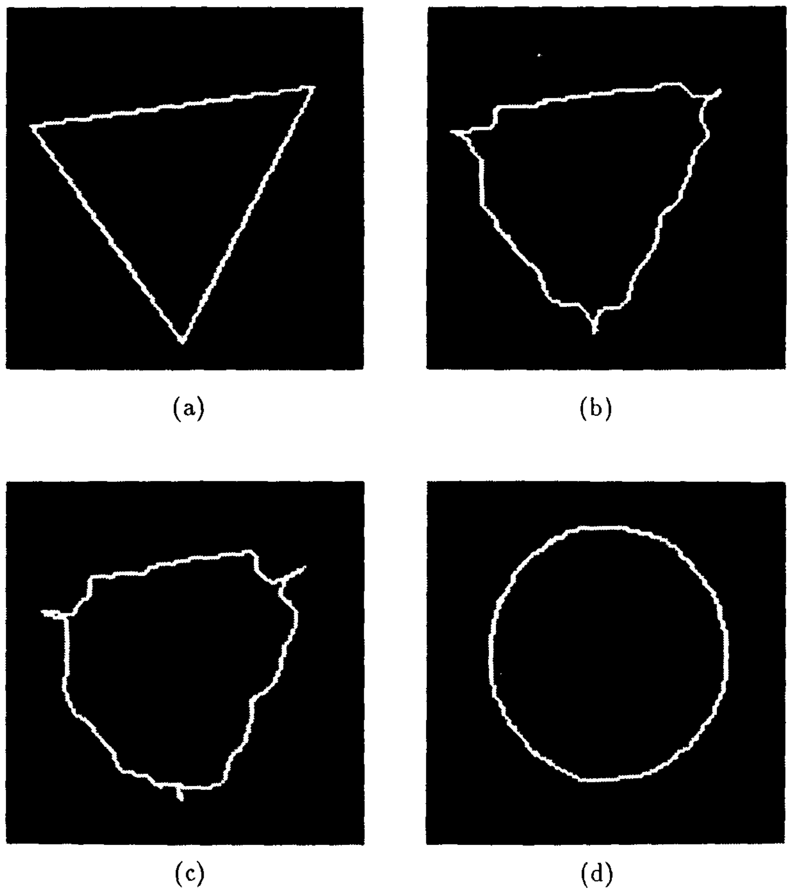


Figure 1: (a) A synthetic image of a circle and the initial position of the curve. (b) The position of the curve after 3 iterations and (c) 7 iterations. (d) The final outline.

At every iteration, we compute the derivative of V with respect to deformations of the contour C:

$$\begin{aligned}\frac{\partial V}{\partial X} &= \frac{\partial F_A}{\partial X} + \frac{\partial F_{\text{grad}}}{\partial X} \\ \frac{\partial V}{\partial Y} &= \frac{\partial F_A}{\partial Y} + \frac{\partial F_{\text{grad}}}{\partial Y} \quad .\end{aligned}$$

In Appendix A, we derive expressions for these derivatives. To perform the optimization, we use a gradient descent technique modified in two respects:

- **Impose a geometric constraint.** The simplest geometric constraint is *smoothness*, which we enforce by incrementing X and Y in the direction of the gradient as in a standard gradient procedure and then recursively smooth the resulting coordinate vectors using the mask $(.25, .5, .25)$. This procedure is fast since it can be implemented using only integer additions and shifts, but no floating point operations or multiplications. To generate the results presented in this paper, at each iteration, the X and Y vectors are convolved 10 times with the mask $(.25, .5, .25)$.

  Experimentally, this procedure yields results nearly identical to the snake method, which replaces the smoothing procedure by simulating the dynamics of a curve moving in a viscous medium. The snake technique is summarized in Appendix B, where we show that our constrained optimization is equivalent to applying the snake to the potential $V - \alpha L^2$, where L is the object boundary and α grows with the amount of smoothing.

  Score optimi'ing curves operating under this constraint will attempt to shrink (or expand) to match the contours of an object and yield a smooth outline. In the application domain of buildings, described below, we instead constrain the contour by fitting a rectilinear polygon to the contour after each iteration.

- **Normalize the derivatives of the score.** The magnitude of the derivatives is not related to the current distance of the contour from its optimal location. Therefore, for every iteration, we pick a step size and retain only the sign of the derivatives which indicates in which direction the contour should move, resulting in a string FX of $-1,0$ and 1 for the X coordinates and a string FY for the Y coordinates. We then normalize the string so that $(||FX||^2 + ||FY||^2)/n = \delta^2$, where n is the number of boundary points and δ the current step size, and replace $\partial F/\partial X$ and $\partial F/\partial Y$ by FX and FY in Eq. (14). This ensures that the displacement of each point is on the average of magnitude δ.

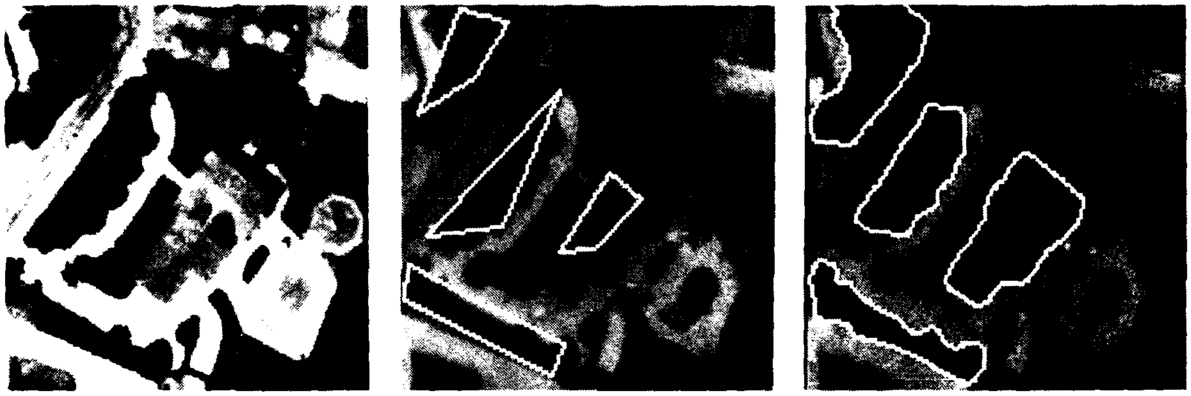


Figure 2: (a) An aerial image of a suburban scene. (b) Interactively entered initial contours. (c) Final outlines after optimization.

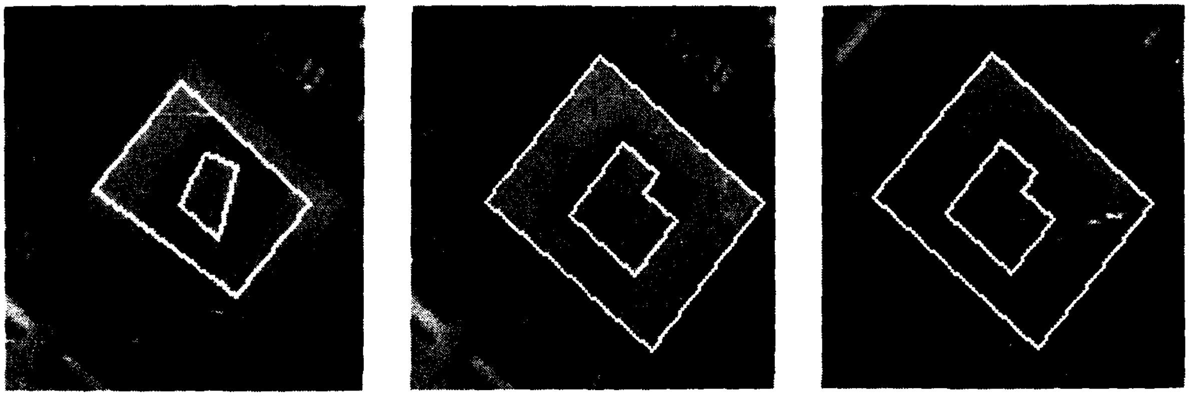


Figure 3: (a) Initial contours in the left image of stereo pair. (b) Final polygonal outlines after optimization. (c) Matching outlines in the right image.

Because of the smoothing terms, deformations are propagated along the curve at every iteration, making this procedure considerably faster than ordinary gradient descent.

Since the objective function is highly nonconvex, after each iteration we recompute the score and verify that it has increased. If, instead, it has decreased, the curve is reset to its previous position and the step size reduced.

The optimization proceeds until the curve stabilizes. For example, going from the initial estimates of the closed curve shown in Figure 1(a) to the final result shown in Figure 1(d) took only 10 iterations. Figures 1(b) and 1(c) show the position of the curve after 3 and 7 iterations respectively.

We now turn to the aerial image in Figure 2(a), the four initial contours shown in Figure 2(b) yield, after optimization, the final outlines of Figure 2(c). Note that the corners of the house are slightly rounded because of the presence of the smoothing term.

Finally, we look at the central building in Figure 3. Given the the two initial contours shown in Figure 3(a), the algorithm generates the outlines shown in Figure 3(b). Using a second image, the elevation of the contours can now be automatically determined by maximizing the stereo effectiveness F_S defined by Eq. (5). For all hypothesized elevations within a given range, the projection of the outlines in the second image and the corresponding value of F_S. are computed. The elevation for which F_S is maximal is the height with the strongest supporting evidence. In Figure 3(c), we show the computed projections of the contours in a second image of the same scene.

In the following sections, we turn from operator-initiated applications to purely automated feature-extraction systems based on the same objective functions.

Automated Building Extraction

The semiautomated system described in the previous section had to be manually cued by the human operator. In this section, we describe a procedure for automatically generating such cues by using a building model consisting of three-dimensional rectilinear outlines conforming to the area, edge, and stereo models of our standard objective function.

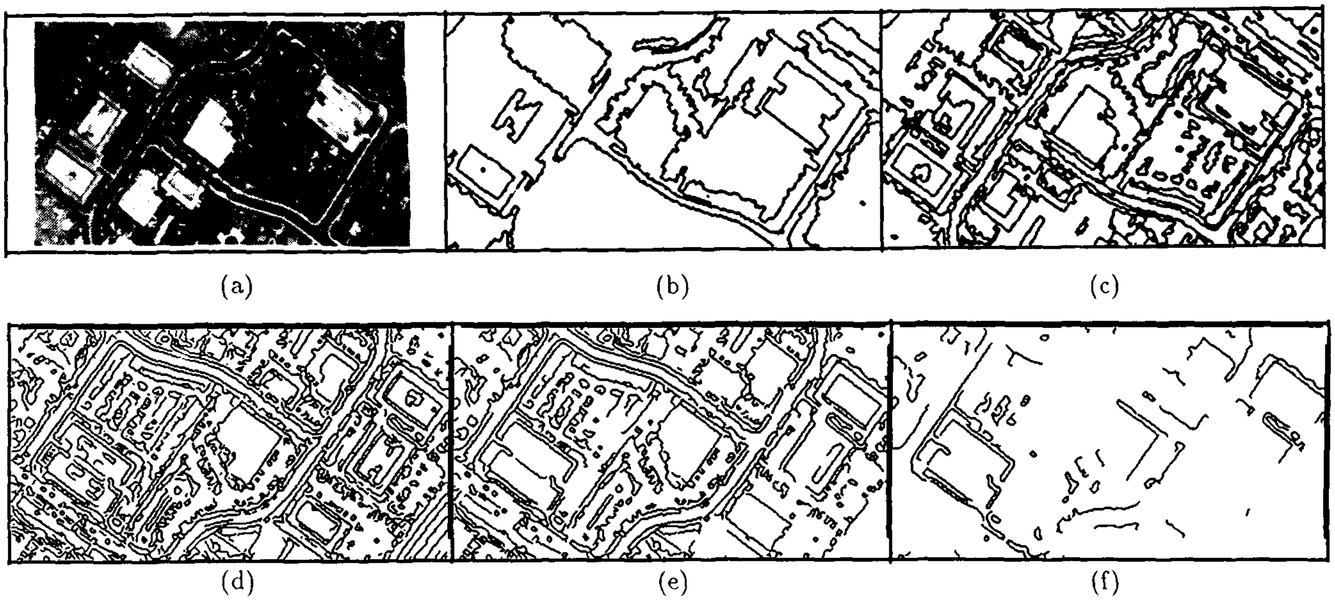


Figure 4: (a) An aerial suburban scene. (b) A Laws segmentation with undersegmented roofs. (c) Oversegmentation resulting from a different parameter choice. (d) (e) (f) Canny edge images computed progressively lower edge-strength thresholds.

The combined use of all three types of evidence is critical for the general success of the approach; if any one of these evidence sources is omitted, the class of interpretable images is markedly restricted.

Details of the Parsing Procedure. The detailed steps in the parsing procedure that we have implemented are outlined below:

- **Generate Edge Cues.** As initial input, our system can use either edge maps or segmentation region boundaries. We have used Canny edge maps [2], the Laws segmenter [17,18], and the Leclerc segmenter [19]. The edge cues are then generated by extracting edge segments with the appropriate geometry from either region boundaries or linked edge pixels. The system builds model edges by finding breakpoints in the initial curves [4] and fitting curves obeying the model geometry (e.g., straight segments for buildings) between those breakpoints. Edge location is optimized using the technique described in Fua and Leclerc [20]. The resulting edges are scored using the edge-quality portion of the objective function; we compute the percentage of edge pixels that are maxima of the local gradient, and retain only the best edges.

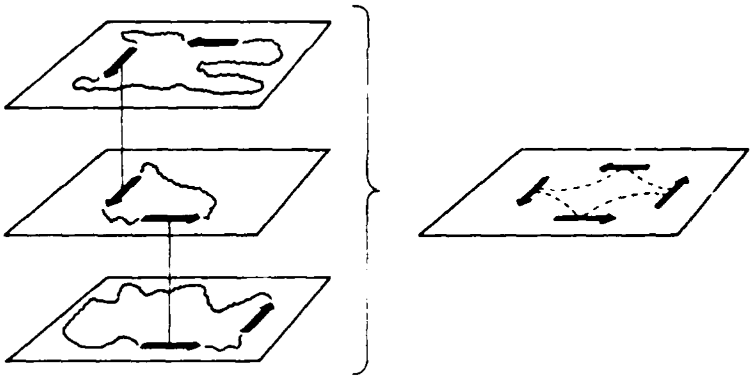


Figure 5: Construction of binary relationships among elementary edge structures across a set of segmentations.

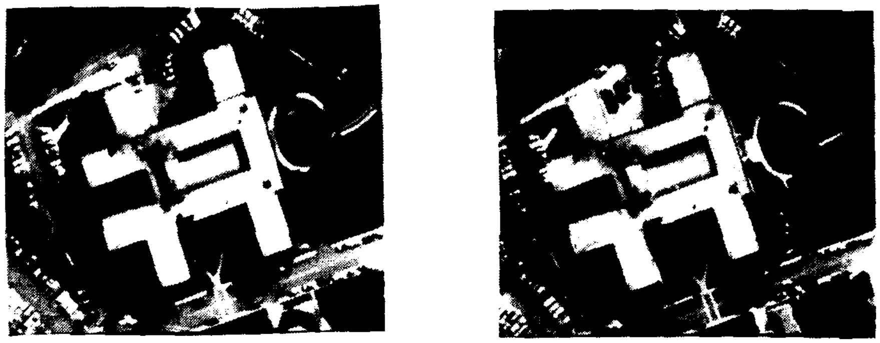


Figure 6: A stereo pair of images containing a large complex building.

Hierarchies. No single parameter setting can be expected to handle all target objects in one image, much less in multiple images. To solve this problem, we use a *parameter hierarchy* of either edge maps or segmentations. Such hierarchies arise naturally when using low-level operators with variable sensitivity parameters.

- When using the Canny map, we compute a series of maps with monotonically increasing edge strength thresholds. We then merge edges of different Canny edge maps into edges into a single list by retaining the best nonoverlapping edges.
- When using segmentation boundaries, we compute a series of segmentations ranging from undersegmentation to oversegmentation, and extract edges from the region boundaries only. We retain each edge's region identification, and, instead of removing overlapping edges in different levels of the hierarchy, we mark them as belonging to the same physical edge.

Shape cues are merged across the hierarchy in subsequent analysis steps to achieve image independence. In Figure 4(a), we show a typical image containing suburban houses. Figures 4(b–e) show portions of both a segmentation parameter hierarchy and an edge map hierarchy to illustrate typical problems such as the absence of relevant shape information and the presence of excess irrelevant information encountered when one attempts to use any single parameter setting.

All the results in this paper were computed using the Canny edge operator [2]. However, we have found that the process can be considerably more efficient if segmentation cues are used because the segmentation region preselects groups of edges adjacent to uniform image regions and reduces the search space.

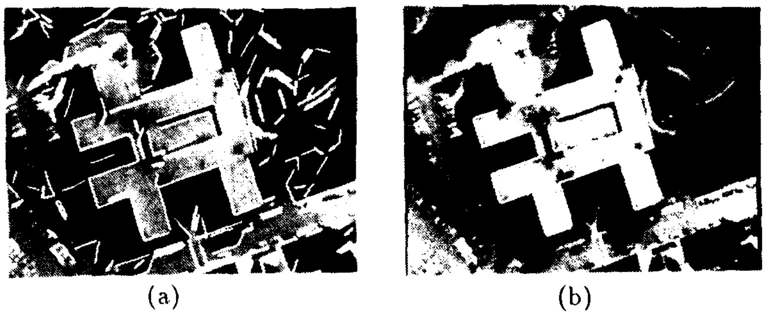

(a) (b)

Figure 7: Steps in the parsing procedure. (a) Straight edges extracted from the original image data. (b) The cluster of egdes corresponding to the building.

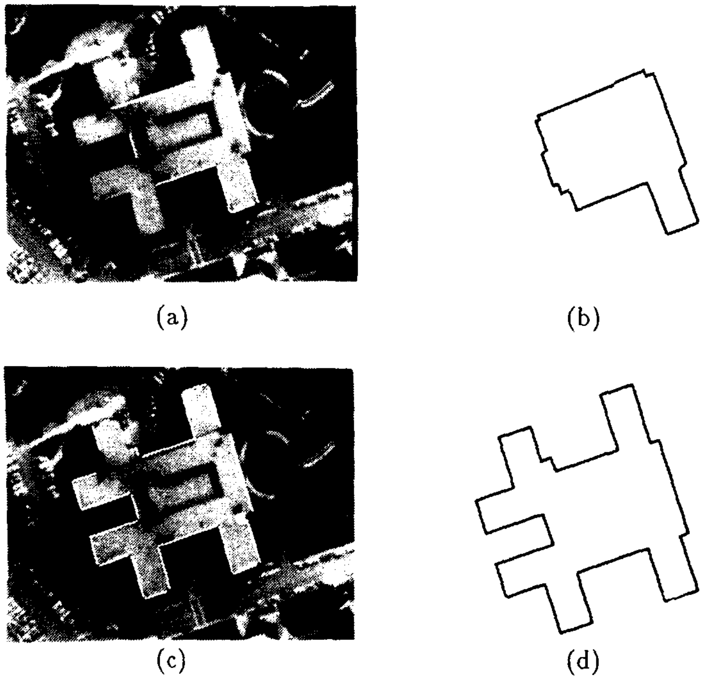

(a) (b) (c) (d)

Figure 8: Steps in the parsing procedure. (a) A cycle of edges suggesting the presence of a good building object. (b) The enclosure resulting from completing the missing links in the cycle. (c) A cycle of edges that has no consistent semantic interpretation. (d) The resulting enclosure.

- **Construct Arcs (Compatible Binary Edge Relationships).** Pairs of edges that satisfy the geometric model constraints are assigned binary edge relationships. If a segmentation is being used to generate cues, we first compute relationships among edges belonging to single regions at one level of the hierarchy; next, we use transitivity to construct binary relationships among edges occurring at completely different levels of the hierarchy, as illustrated in Figure 5. When the regions correspond to good cycles, using the segmentation serves to make cycle construction more reliable and efficient.

  Finally, a patch in the image is associated with each detected binary structure. The area component of the total objective function score for the patch is computed, and only structures with a high score are retained.

- **Construct Cycles.** We next gather all the *clusters* of edges related by chains of binary relationships; related edges are clustered together based on their having similar photometry. Within the clusters , we look for *cycles*, which we define as circular lists of binary relationships satisfying both the photometric and geometric model requirements. We illustrate cycle construction by examining the images in Figure 6. Figure 7 shows the edges and two examples of graphs constructed from them. Figures 8(a) and 8(b) illustrate a building cycle and the enclosure resulting from running the model-based completion algorithm. Figures 8(c) and 8(d) show the results for a more typical cycle; this undesired hypothesis illustrates the great need in such a system for a way to distinguish good hypotheses from nonsensical ones.

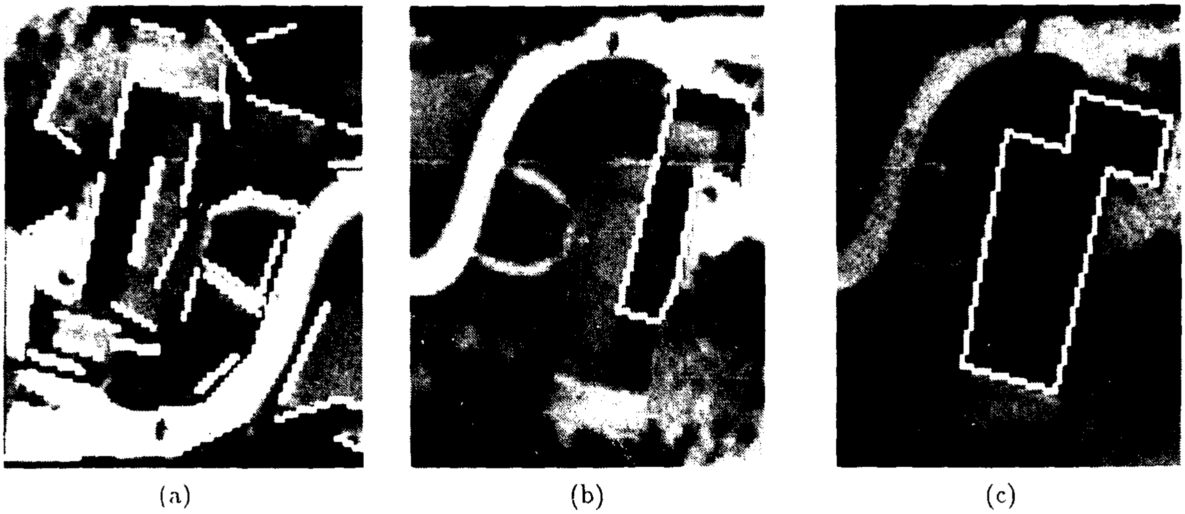


Figure 9: (a) Bare edges in an image containing a building. (b) A closed cycle before optimization, including only a portion of the building. (c) The cycle after optimization has expanded to fill in the building areas matching the model.

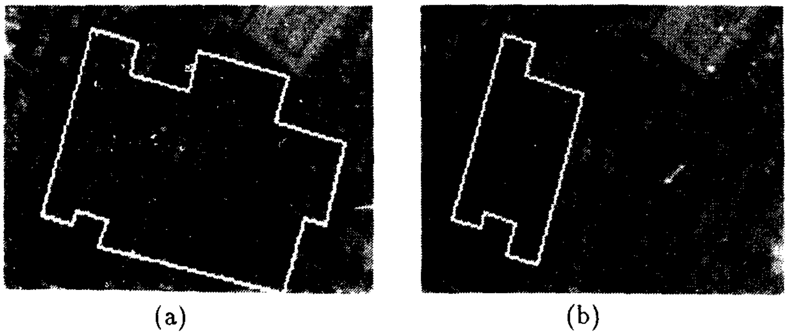


Figure 10: Two enclosures generated by the system. The larger one, (a), was built using spurious edges that accidentally lined up with the true building, (b). At small scale, hypothesis (a) dominates because, although it fits the photometric model less well, it is much larger.

- **Build Enclosures.** To form enclosures, the gaps in the cycles of edges are first closed using a procedure that finds the best rectilinear path. The closures are found using a variant of the F^* algorithm [5]. The resultant closed contour forms a *building cue* that is optimized using the method of the previous section.This optimization stage is crucial in several respects
  - **Compensation for Poor Photometry.** Optimization moves the contour to a local maximum of the objective function. However, this maximum can be an hallucination that conforms to the geometry but has poor photometric evidence. In Figure 9, we show how the optimization of a deficient hypothesis can

produce a much-improved building candidate. In the case when neighboring regions have low contrast, the system can hallucinate enclosures with good area scores that span both regions, as shown in Figure 10. When such hallucinations do occur, edge quality is an important discriminatory tool.

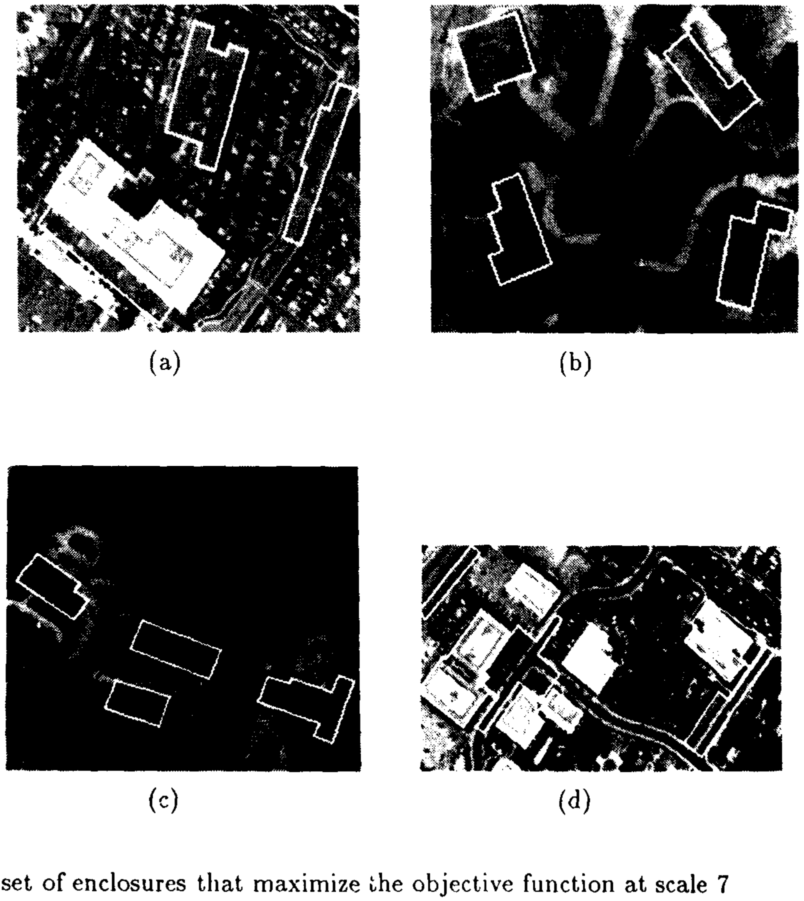
- **Collapse of Multiple Hypotheses.** The cycle builder typically generates massively redundant hypotheses with overlapping common portions. Cycles with sufficient overlap will be optimized to identical enclosures, thus serving to reduce the redundancy of the hypotheses to be considered. This strategy is extremely effective in finding such things as faint building edges that an edge detector may miss, but that are paired with a strong edge on a different side of the building.

- **Find Stereo Match.** When stereo is being utilized, the system computes the parameters of the three-dimensional plane describing the world-position of the enclosure by optimizing the stereo effectiveness measure.

- **Select Enclosures.** The current implementation of the system assumes that we are dealing with independent model candidates, which are then scored using the objective functions given in Eqs. (1-6). For complex scenes, the system builds a set of overlapping, and therefore conflicting, enclosures that encompass the candidate model instances mentioned earlier. The system chooses a subset of nonoverlapping enclosures that maximizes the appropriate objective function.

The strength of this procedure is that it integrates model knowledge into every step of the parsing process, while referring continuously back to the image data for confirmation of each hypothesis.

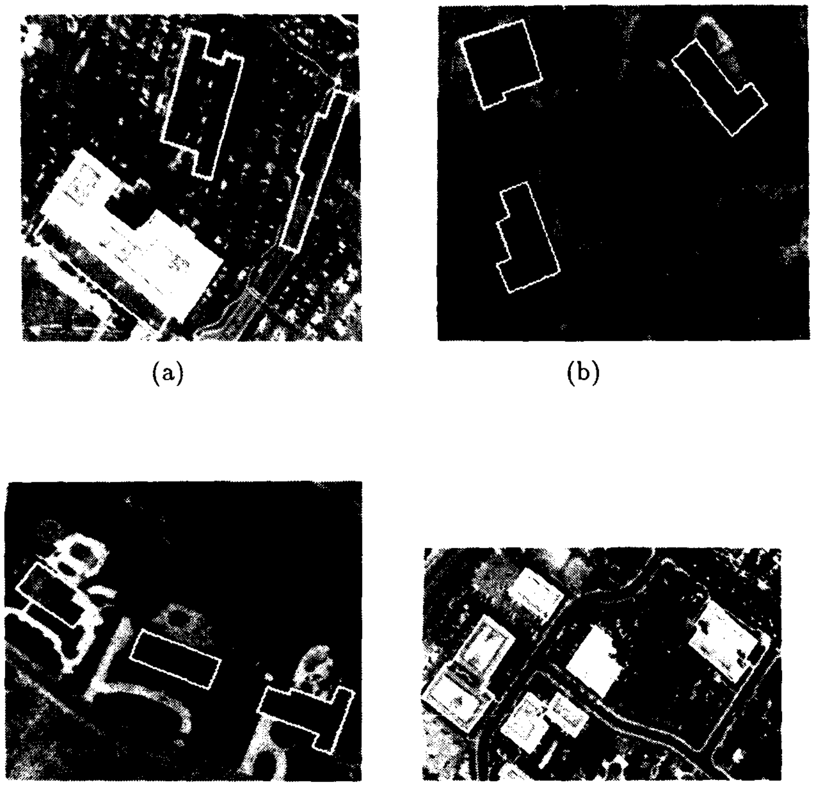
Experimental Results on Buildings


(a) (b)

(c) (d)

Figure 11: Subset of enclosures that maximize the objective function at scale 7

We now show the results of running the system on a series of complex images with very different photometry. Using the basic building model outlined in an earlier section, the hypothesis generator automatically produces several



(a) (b)

(c) (d)

Figure 12: Subset of enclosures that maximize the objective function at scale 8

hundred model candidates in each image. The objective function ranks theses hypotheses according to their scale-dependent score. The scale parameter is the only parameter we have varied from image to image. Since the scale has a semantic meaning, it is inevitable that a semantic decision to select its value will be required at this stage to achieve the performance goals set by the human operator. We have made no attempt to encode the knowledge needed to automate the selection of scale.

Buildings. In the companion paper [9], we have argued that the scale parameter s that appears in the objective function effectively fixes the lower bound on the geometric quality and size of aceptable structures. In Figures 11 and 12 we compare the results of the selection procedure of four different images, once with scale 7, and again with scale 8. At scale 7, all the buildings are picked out, but some candidates with marginal characteristics such as yards and parking lots are retained. At scale 8, the rejection ratio of spurious candidates is improved, but now some legitimate buildings are also lost. If stereo information were available, we could use it to reject some of the spurious candidates.

Stereoscopic Buildings. When stereoscopic or multiple imagery such as Figure 6 is available, the ambiguities inherent in the identification of rectilinear, building-like objects in monoscopic imagery are largely resolved.

In Fig. 13 we show the outline of the three highest-scoring building candidates found by the system and their relative scores. Note that the incomplete building shown in Fig. 13 (a) has very uniform photometry, while the perceptually correct outline shown in Fig. 13(c) includes large shadows and darker pixels that degrade its area-encoding effectiveness. As a result that the scores (using only the edge and area terms of the objective function) of the two outlines are extremely close. Not surprisingly, the Laws segmenter [17,18] produces the segmentation shown in Fig. 14 in which a region very similar to the erroneous outline has been extracted.

However, when we project the contours found in the left image into the right image and take the corresponding stereo effectiveness into account, the score of the "correct" parse becomes considerably larger than the score of the

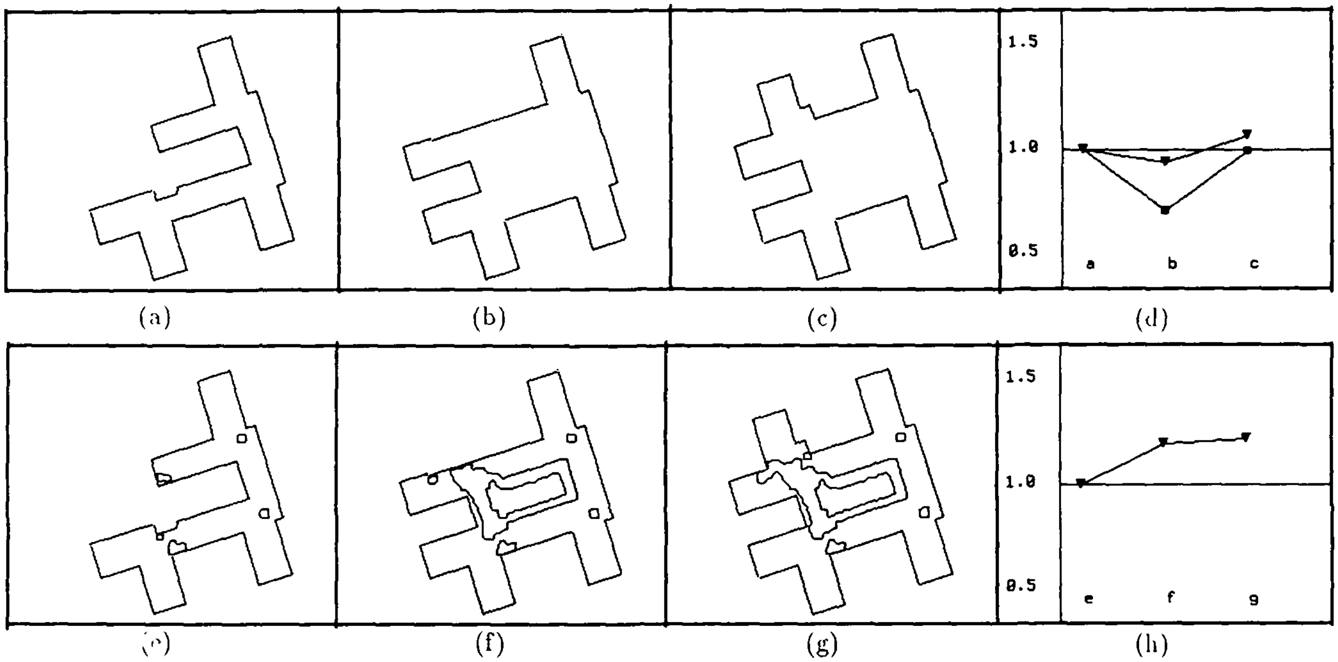


Figure 13: (a) (b) (c) The three highest scoring hypotheses generated by the system when parsing the left image of Fig. 6. (d) Ratio of the scores at scale 8 of (b) and (c) to the score of (a) when using stereo information (triangles) or not using it (squares). Without stereo (a) and (c) have similar scores, while (c) dominates whith stereo. (e) (f) (g) (h) The same three hypotheses with holes and the ratio of their scores including stereo. (g) dominates even more clearly and (f) in intermediate between (e) and (g).

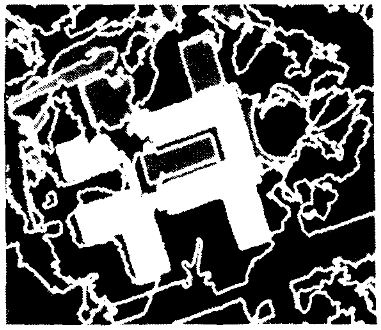


Figure 14: A Laws segmentation with building region highlighted

erroneous one. Thus, in a case like this one, the additional stereoscopic information helps the system overcome ambiguities that arise in complex scenes. Upon optimizing the stereo effectiveness, we know the three-dimensional position of each rectilinear contour; we can thus produce three-dimensional objects having the observed two-dimensional upper surface. When we supply the best candidate (Fig. 13) to the SRI cartographic modeling system [13,14], we can generate synthetic three-dimensional views of the scene such as the ones shown in Figure 15.

Effect of Improving the Anomaly Model. To further disambiguate difficult situations such as the one described in Fig. 13, we can also extend our model to allow object contours with holes encoded solely by their boundary encoding cost as shown in Fig. 13 (e) (f) and (g). When using both this more sophisticated model and the stereographic information, the ranking of the three model instances plotted in Fig. 13 (h) is now closer to that of a human. Thus,

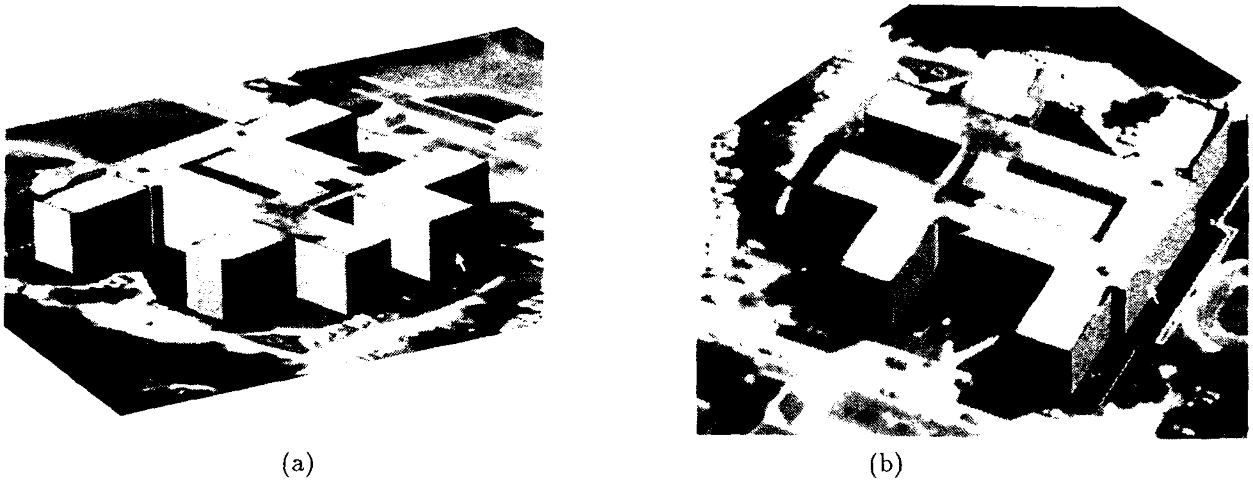


Figure 15: Synthetic views of a three-dimensional scene with the original image texture mapped onto the central building model. Scene areas that cannot be texture-mapped because they are not visible from the original camera position are shown in black.

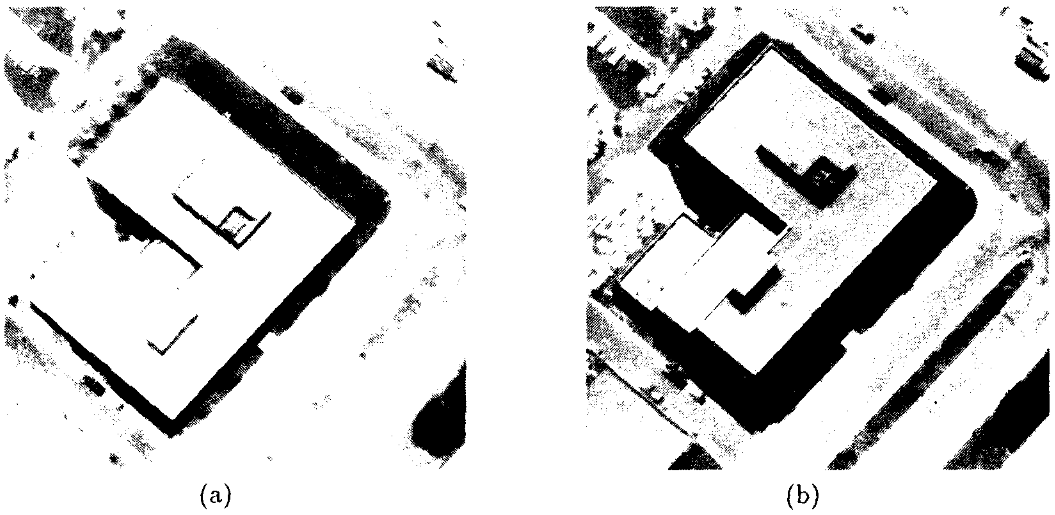


Figure 16: (a) Building candidates found in the left image. (b) Building candidates projected to the right image.

more sophisticated modeling techniques as well as additional photometric evidence help produce more reliable scene parses.

As an illustration of the full stereo parse, we show in Figure 16 a multitiered building with the best subset of stereo enclosures that appear to be more than twenty feet high, and in Figure 17, an oblique view synthesized using the resulting parse.

Conclusions

In this work, we have proposed a feature-extraction approach that uses a probabilistic objective measure as the heart of either operator-initiated feature-extraction procedures, or a totally automated heuristic hypothesis-generator tuned to produce a spectrum of local minima of the objective function. The step of the automated system

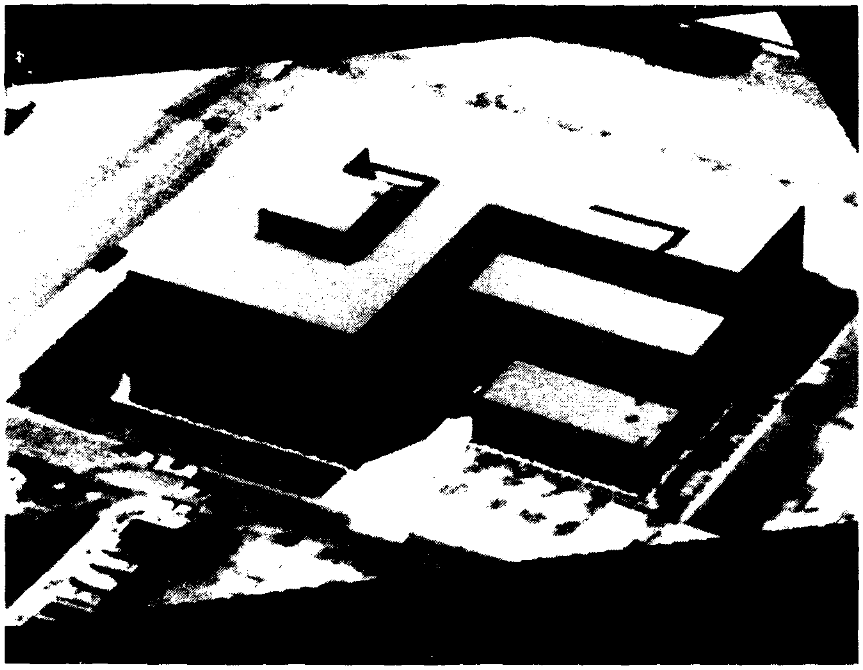


Figure 17: Synthetic view of a three-dimensional scene with the original tiered building image texture mapped onto the building model.

just preceding final ranking includes a local optimization identical to the operator-initiated process. These techniques unify the optimization requirements of low-level photometric evidence with high-level heuristic or operator-imposed semantic constraints.

The system successfully discovers a high proportion of building-like objects in aerial images with difficult photometry; such images are likely to cause standard image-partition processes to miss many or most object instances. Stereoscopic information is found to be particularly effective in resolving building candidates.

In future work, we plan to extend the range and complexity of the models supported, to include three-dimensional modeling and illumination information, and to pursue a comprehensive treatment of high-level interdependencies among models to support complex cartographic analysis requirements.

Appendix A: Derivatives of the Effectiveness

Derivatives of the area term

To estimate the derivatives of F_A, we first compute the contribution dF_A of every point (x,y) in the image when added to the patch defined by C. We recall that

$$\begin{aligned} F_A &= (8-k_A)\frac{A}{s^2} \\ k_A &= n(\log\sigma + c) + 8\overline{n} + \left[n\log\frac{n}{A} + \overline{n}\log\frac{\overline{n}}{A}\right], \end{aligned}$$

where $c = \frac{1}{2}\log(2\pi e)$ and n and $\overline{n}$ are the numbers of normal and anomalous pixels, respectively. which we can rewrite as:

$$k_A = n(c_1 - \frac{\log v}{2}) + n\log n + \overline{n}\log\overline{n} - A\log A,$$

where $c_1 = 8.0 - c$ and $v = \sigma^2$. To evaluate the contribution of an individual pixel we must distinguish two different cases:

1. The pixel's deviation d from the planar fit lies in the main gaussian peak. In that case, n and A must be incremented by 1, while the the overall variance v is modified by $dv \approx (d^2 - v)/n$. Therefore dF_A can be computed as follows:

$$\begin{aligned} dF_A &= (c_1 - \frac{\log v}{2}) - \frac{c_2}{2} n\frac{dv}{v} + \log n - \log A \\ &= (c_1 - \frac{\log v}{2}) - \frac{c_2}{2}(\frac{d^2}{v} - 1) + \log n - \log A \end{aligned}$$

   where $c_2 = \log_e 2$.

2. The pixel does not belong to the main peak. Its contribution to $\overline{n}$ and dF_A can then be taken as

$$dF_A = \log\overline{n} - \log A.$$

Having computed dF_A, we can now estimate $\partial F_A/\partial X$ using finite differences. Let us consider a boundary point $P = (x,y)$. Our implementation assumes that the boundary points themselves do not belong to the patch. There are four possible patterns for the 3×1 horizontal neighborhood centered around P:

Case a:	1	P	0
Case b:	0	P	1
Case c:	1	P	1
Case d:	0	P	0

where 0 represents a point that does not belong to the patch and 1 represents a point that does.

- Case a: If P moves to the right, the center point is added to the patch and F_A becomes $F_A + dF_A(x,y)$; conversely if P moves to the left, the left point is removed from the patch and the F_A becomes $F_A - dF_A(x-1,y)$. $\partial F_A/\partial x$ is therefore estimated to be
$$\frac{\partial F_A}{\partial x} = +\frac{dF_A(x,y) + dF_A(x-1,y)}{2}.$$
- Case b: Similarly
$$\frac{\partial F_A}{\partial x} = -\frac{dF_A(x,y) + dF_A(x+1,y)}{2}.$$
- Case c and d: The boundary is locally horizontal,
$$\frac{\partial F_A}{\partial x} = 0.$$

$\partial F_A/\partial X$ is the vector of the $\partial F_A/\partial x$ for all the points in C. $\partial F_A/\partial Y$ is computed similarly by replacing horizontal neighborhoods by vertical ones. Note tnat dF_A can be computed on a pixel per pixel basis and therefore in parallel for all pixels in the image.

Derivatives of the edge term

Referring to Eq. (10), we write F_{grad} as

$$F_{\text{grad}} = \frac{1}{s} \sum_{C(x,y)} \log \frac{g(x,y)}{g_0}.$$

Here g_0 is the minimum gradient threshold required for an edge to be considered. In practice, we precompute, once and for all, the quantity Γ defined by

$$\Gamma(x,y) = \begin{cases} \log(g(x,y)/g_0) & \text{if } s > g_0 \\ 0 & \text{otherwise} \end{cases} .$$

We also precompute the derivative of Γ, $\partial\Gamma/\partial x$ and $\partial\Gamma/\partial y$. At each iteration, $\partial F_E/\partial X$ and $\partial F_E/\partial Y$ are simply the vectors whose components are the values of $\partial\Gamma/\partial x$ and $\partial\Gamma/\partial y$ at the current boundary points.

Appendix B: Snake Dynamics

Following Terzopoulos [29], we consider a "snake" curve C to be a physical curve defined by the vector (X, Y), embedded in a medium of viscosity $\alpha = 1/\lambda$ and moving under the influence of the potential $V = L^2 - \alpha F$. L^2, the square length of the boundary, can be computed as

$$L^2 = \frac{1}{2} XKX + \frac{1}{2} YKY \ , \tag{11}$$

where K is the tridiagonal matrix with coefficients $-1, 2, -1$. At every iteration of the optimization, we then solve the equation of dynamics,

$$\frac{\partial V}{\partial C} + \alpha \frac{dC}{dt} = 0, \tag{12}$$

where $\partial V/\partial C$ is the vector $(\partial V/\partial X, \partial V/\partial Y)$. Since the deformation energy L^2 in Eq. (11) is quadratic, its derivatives with respect to X and Y are linear. Thus, each iteration of the optimization amounts to solving the two linear equations:

$$\begin{aligned} KX_t + \alpha(X_t - X_{t-1}) &= \alpha \left.\frac{\partial F}{\partial X}\right|_{C_{t-1}} \\ KY_t + \alpha(Y_t - Y_{t-1}) &= \alpha \left.\frac{\partial F}{\partial Y}\right|_{C_{t-1}} \ . \end{aligned} \tag{13}$$

Let $M = (I + \frac{1}{\alpha}K)^{-1}$, Eq. (13) can be rewritten as

$$\begin{aligned} X_t &= M(X_{t-1} + \left.\frac{\partial F}{\partial X}\right|_{C_{t-1}}) \\ Y_t &= M(Y_{t-1} + \left.\frac{\partial F}{\partial Y}\right|_{C_{t-1}}) \ . \end{aligned} \tag{14}$$

For α large enough (typically $\alpha > .01$), the matrix M can be approximated with excellent accuracy by an n-diagonal matrix. We can therefore solve Eq. (14) simultaneously for X and Y by convolving the right-hand terms $X + \partial F/\partial X$ and $Y + \partial F/\partial Y$ with the appropriate mask. In this formulation, the value of α determines the width of the mask and how much X and Y are smoothed – the smaller α, the more smoothing.

The recursive smoothing procedure proposed in the main text amounts to multiplying X and Y with a convolution matrix with coefficients that are centered around the diagonal and are similar to those of the matrix M used to solve the snake equations. The smoothing procedure can therefore be considered an approximation to the snake method ; in practice both techniques yield indistinguishable results.

*

References

[1] D.H. Ballard, "Generalizing the Hough Transform to Detect Arbitrary Shapes," Pattern Recognition, **13**, pp. 111-122 (1981).

[2] J. Canny, "A Computational Approach to Edge Detection," IEEE Trans. PAMI **8**, pp. 679-698 (1986).

[3] J.A. Feldman and Y. Yakimovsky, "Decision Theory and Artificial Intelligence: I. A Semantics-Based Region Analyzer," Artificial Intelligence **5**, pp., 349-371 (1974).

[4] M.A. Fischler and R.C. Bolles, "Perceptual Organization and Curve Partitioning," IEEE Pat. Anal. Mach. Intell. **8**, pp. 100-105 (1986).

[5] M.A. Fischler, J.M. Tenenbaum, and H.C. Wolf, "Detection of Roads and Linear Structures in Low-Resolution Aerial Imagery Using a Multisource Knowledge Integration Technique," Computer Graphics and Image Processing **15**, pp. 201-223 (1981).

[6] P. Fua and A.J. Hanson, "Using Generic Geometric Models for Intelligent Shape Extraction," Proceedings of the AAAI Sixth National Conference on Artificial Intelligence, pp. 706-711 (July 1987).

[7] P. Fua and A.J. Hanson, "Extracting Generic Shapes Using Model-Driven Optimization," Proceedings of the Image Understanding Workshop, Boston, MA, pp. 994-1004 (April 1988).

[8] P. Fua, "Object Delineation as an Optimization Problem, A Connection Machine Implementation," in Proceedings of *AI Application of Supercomputers: The Vision Problem*, Fourth International Conference on Supercomputing, Santa Clara, CA (30 April - 5 May 1989).

[9] P. Fua and A.J. Hanson, "Objective Functions for Feature Discrimination: Theory," in this Proceedings, Proceedings of the Image Understanding Workshop, Palo Alto, CA (May 1989).

[10] P. Fua and Y.G. Leclerc, "Model Driven Edge Detection," Machine Vision and Applications, *in press*.

[11] M.P. Georgeff and C.S. Wallace, "A General Selection Criterion for Inductive Inference," in Proceedings of Advances in Artificial Intelligence, Pisa , Italy Sept. 1984, T. O'Shea (Ed.) (North Holland, Amsterdam, 1984).

[12] R.W. Hamming, "Coding and Information Theory," Prentice Hall, New Jersey, (1985).

[13] A.J. Hanson, A.P. Pentland, and L.H. Quam "Design of a Prototype Interactive Cartographic Display and Analysis Environment," Proceedings of the Image Understanding Workshop, pp. 475-482 (February 1987).

[14] A.J. Hanson and L. Quam, "Overview of the SRI Cartographic Modeling Environment," in Proceedings of the Image Understanding Workshop, Boston, MA, pp. 576-582 (April 1988).

[15] R.M. Haralick, "Digital Step Edges from Zero Crossings of Second Directional Derivatives," IEEE Trans. PAMI 6, pp. 58-68 (1984).

[16] M. Kass, A. Witkin,D. Terzopoulos (1988) "Snakes: active contour models," International Journal of Computer Vision, (1)(4):321-331.

[17] K.I. Laws, "Goal-Directed Texture Segmentation," Technical Note 334, Artificial Intelligence Center, SRI International, Menlo Park, California (September 1984).

[18] K.I. Laws, "Integrated Split/Merge Image Segmentation," Technical Note 441, Artificial Intelligence Center, SRI International, Menlo Park, California (July 1988).

[19] Y.G. Leclerc, "Image Partitioning as Constructing Simple Stable Descriptions," Proceedings of the Image Understanding Workshop, Boston, MA (April, 1988); "Image Partitioning as Constructing Simple Stable Descriptions," to be published in International Journal of Computer Vision.

[20] Y.G. Leclerc and P. Fua, "Finding Object Boundaries Using Guided Gradient Ascent," Proceedings of the the 1987 Topical Meeting on Machine Vision, Lake Tahoe, California, pp. 168-171 (March 1987). Also presented at the DARPA Image Understanding Workshop, Los Angeles, California, pp. 888-891 (February 1987).

[21] D.M. McKeown and J.L. Denlinger, "Map-guided feature extraction from aerial imagery," Proc. 2nd IEEE Workshop on Computer Vision, pp. 205–213 (May, 1984).

[22] D. McKeown, W.A. Harvey, and J. McDermott, "Rule-Based Interpretation of Aerial Imagery," IEEE Trans. PAMI **7**, pp. 570–585 (1985).

[23] A. Huertas and R. Nevatia, "Detecting Buildings in Aerial Imagery," Computer Vision, Graphics and Image Processing **41**, pp. 131–152 (1988).

[24] Y. Ohta, T. Kanade, and T. Sakai, "A Production System for Region Analysis," Proc. 6th Inter. Joint Conf. on Artif. Intell., pp. 684–686 (1979).

[25] J. Rissanen, "A Universal Prior for Integers and Estimation by Minimum Description Length," The Annals of Statistics **2**, pp. 416–431 (1983).

[26] J. Rissanen, "Minimum-Description-Length Principle," in *Encyclopedia of Statistical Sciences*, **5**, pp. 523-527, (1987).

[27] A. Rosenfeld, "A Nonlinear Edge Detection Technique," Proc. IEEE, **58**. pp. 814–816 (1970).

[28] G. Schwarz, "Estimating the Dimension of a Model," The Annals of Statistics **6**, pp. 461–464 (1978).

[29] D. Terzopoulos, "On Matching Deformable Models to Images," Topical Meeting on Machine Vision, Technical Digest Series, Optical Society of America, Washington, D.C., (12):160–167 (1987).

REAL-TIME MOTION TRACKING USING SPATIO-TEMPORAL FILTERS

Peter K. Allen

Department of Computer Science
Columbia University
New York, New York 10027

Abstract

One of the main impediments to the use of machine vision techniques in robotic applications has been the lack of real-time response. Many of the more appealing motion detection algorithms contain a burdensome computational cost which precludes real-time implementation. However, the advent of new high speed architectures for image processing has opened up the possibility of real-time motion detection and quantification. This paper discusses the use of spatio-temporal filtering on a frame-rate video processor to perform real-time motion tracking with a camera mounted on a robotic arm.

INTRODUCTION

Motion detection and quantification in machine generated imagery has been studied quite extensively over the last few years. One of the goals of this work has been to determine optical flow fields that measure image velocity at each pixel in the image. A variety of techniques have been used with varying results including, among others, matching based techniques [3,6,17], gradient based techniques [7,11,16] and spatio-temporal energy methods [1,10].

Our interest in this area is in exploiting motion tracking for robotic applications. In particular, we wish to track moving objects (parts on a conveyor or on a rotating table for example) with an arm mounted camera in order to pick them up with a stable grasp using a dexterous multi-fingered Utah-MIT hand [2]. Applying these methods to real-time motion tracking is, however, difficult. The computational demands of these methods, usually carried out on a pixel by pixel basis, put a large computational burden on any system in which real-time response (for servoing of an arm mounted camera) is required. Previous work by Burt et al. [5] has focused on high speed feature detection and hierarchical scaling of images in order to meet the real-time demands of surveillance and other robotic applications. Related work has been reported by Lee and Wohn [13] and Wiklund and Granlund [19] who use image differencing methods to track motion. Corke, Paul and Wohn [8] report a feature based tracking method that uses special purpose hardware to drive a servo-controller of an arm-mounted camera. Goldenberg et al [9] have developed a method that uses temporal filtering with similar hardware to our own. Luo, Mullen and Wessel [14] report a real-time implementation of motion tracking in 1-D based on Horn and Schunk's method.

We have exploited the properties of the PIPE [4,12] real-time image processing machine in this work to achieve real-time tracking of objects in 2-D using a combination of feature based spatial and temporal processing . The motion energy in a scene is identified and used to control the arm mounted camera, keeping the moving object centered in the field of view.

In our initial work, we are using the experimental setup shown in figure 1, which consists of a CCD camera mounted on the end effector of an IBM 7575 SCARA arm. The IBM arm is controlled by a PC based controller (Figure 2) that allows user access to the low level control of the arm. In our experimental environment, we have a single moving object on a homogeneous worktable, similai to an industrial setting. The goal is to track the motion of the moving object in the scene and keep the objec. in the center of the field of view. The motion is very arbitrary, generated by a toy mobile robot that "bounces" off obstacles in the scene and changes its direction of motion arbitrarily i.e. non-predictive), thus precluding us from using predictive motion or Kalman filter tracking techniques.

Thus, we do not need to be able to compute a full optic-flow field at each pixel in the image, but rather, we need to detect where the object is and quantify its global motion (translation) in order to move the hand for grasping. Since we wish to track the motion with an arm mounted camera we are faced with being to able to update the robot arm parameters at high enough rates to keep the object centered in the field of view of the imaging system attached to the robot arm. Our work is notable in that we are explicitly using spatio-temporal processing at real-time rates in conjunction with robotic arm control. The technique has proved to be quite good at maintaining the moving object in the central field of view of the arm-mounted camera system.

ARCHITECTURE OF THE PIPE IMAGE PROCESSOR

In tasks such as tracking a moving part and picking it up, vision sensing has to be synchronized with robot motion in real-time. Typically, the processing of images cannot keep up with the servo rates of joints in an arm, causing delays in movement. To alleviate this, we are using a PIPE image processing system which can process a single image in one video field time. Each stage in the system, called a Modular Processing Stage (MPS), is designed so that all input, processing, and output are completely synchronous with the video-raster and thus allows a complete image to be treated as one data structure. The stages of the pipeline are connected along a number of different pathways. There is a forward path connecting the output of each stage to the input of the next stage, a backward path connecting the output of each stage to the input of the previous stage and a recursive path connecting the output of each stage to its own input. In addition, there are six video buses to connect the output of any stage to the input of any other. Each of these data-paths is eight-bits wide. Images can be made to stream between stages, spending one cycle (1/60th of a second) in each stage for processing. The hardware modules available in each stage include:

- Two image buffers, 256x256x8 bits each, for storing images.
- Two neighborhood operators to do any arbitrary 3x3 or 9x1 convolution on the complete image.
- Four look-up table operators to do any arbitrary point transformation operation on the complete image, such as multiplying each pixel in the image by 2.
- Three ALU's to do simple operations on two images, such as subtracting one image from the other, pixel by pixel.
- A Two Valued Function Module (TVF), that is a very powerful tool to do any arbitrary operation on two images, or to perform arbitrary image warping operation operations, such as rotating an image by an arbitrary angle.

It needs to be emphasized that all the operations in a stage can be done on the complete image, and can be finished in 1/60th of a second. Given this capability, we have developed a number of algorithms to implement real-time vision modules with the goal of fast and robust motion determination.

SPATIO-TEMPORAL FILTERING FOR MOTION DETECTION

A robust but computationally demanding approach to motion detection and quantification is to use a number of spatio-temporal filters that are "tuned" to different discrete image velocities [1]. By finding peaks in the responses of this filter suite, the velocity of an image event can be estimated. The number of filters needed for such a method is prohibitive for a real-time application such as ours. Our method instead lies upon a single spatio-temporal filter that will isolate motion energy in the scene quickly, but will not be as robust in determining the actual velocity in the scene. However, since we are tracking a single moving object, our arm servoing procedure can effectively track the motion that has been isolated in the scene.

A simple technique that can isolate motion energy is to difference two successive images in the scene. This techniques suffer from a number of problems that makes it less than robust. Among these problems are noise and shifting image boundaries in a stable digital image, aliasing effects due to texture on the object and field alignment problems. The effects of any differencing method must be smoothed both spatially and temporally for meaningful motion isolation.

Our attempt to track moving objects in real-time begins with the implementation of a spatio-temporal filter. In spatial image processing, an edge detection method due to Marr and Hildreth [15] is to calculate the Laplacian of a Gaussian filtered image. The Gaussian part of this separable operation is used to smooth the image, after which the Laplacian operator generates essentially a second derivative response in the smoothed image. The zero-crossings of this second derivative operator are then used to find edges in the image. An analog of this operator in the temporal domain can be approximated by taking a second derivative operation on a spatially smoothed input image. The temporal operator can be extended over a number of successive frames and zero-crossings isolated from it. To implement suc'h a filter on the PIPE, three successive images (using every other field for spatial consistency) are spatially smoothed with a Gaussian filter (note: more images in the temporal dimension could be used with a correspondingly longer latency in the processing cycle). The three images are then temporally filtered on a per pixel basis using a mask that finds second derivatives over three images. The forward, backward and recursive paths on the PIPE allow the spatial processing to be carried out in three successive stages and then the three consecutive spatially filtered images can be combined via the three paths described above. This concurrency is one of the key aspects of real-time implementation on the PIPE

Once the second derivative response is calculated, the zero-crossings can be isolated using a binary neighborhood operator that looks for sign changes across a pixel's 3 x 3 neighborhood. The zero-crossings are somewhat noisy, with oscillatory crossings in areas of weak temporal edge strength. Therefore, a simple energy threshold is used to insure that zero-crossings found by the algorithm are not noise artifacts. The algorithm takes 2 PIPE cycles, which provides a new set of zero-crossings every 1/30th sec. with a latency of 6 PIPE cycles (1/10th sec.).

QUANTIFYING THE MOTION

Once the regions of motion have been found using the spatio-temporal filters discussed above, the motion has to be quantified. In this procedure, a global centroid of the spatio-temporal edges is used to quantify the motion, which can be done in two ways on the PIPE. The first method is pyramid based, using a variation of a method discussed by Goldenberg et al. [9]. Each buffer within an MPS of the PIPE can be "squeezed" (subsampled) by a factor of 2 in each dimension at a cycle boundary, thus allowing a 256 x 256 image to be reduced to a 1 x 1 image in 16 cycles (16/60th of a second). The centroid algorithm operates by finding a local centroid of each 2 x 2 rectangular window of the image, and then squeezing the image. This is repeated 8 times after which the global centroid value is left as the 1 x 1 image. Local centroids are computed by using the spatio-temporally filtered edges derived above as binary blobs that contain the image's motion energy. The binary image of spatio-temporal edges is then ANDed w'.h a ramp image of X pixel coordinates and stored in a MPS buffer. A similar procedure is done for Y axis coordinates of the thresholded binary region. Each of the two images now contain a spatially indexed set of pixels in areas where a blob exists. Using a local 2 x 2 convolution, each 2 x 2 window in the image has a local centroid computed. At the end of this cycle, (performed in 1/60th sec.) the image is squeezed and the process repeated, until a global centroid results in a 1 x 1 neighborhood of the image.

The calculation of centroids of motion energy using this pyramid approach is slow in that it takes 16 cycles to reduce the image to a 1 x 1 image containing the centroid in either the X or Y dimension. Our second implementation takes advantage of a special purpose hardware board, ISMAP (iconic to symbolic map). The ISMAP works by forming accumulator totals in one cycle and a simple division operator allows the centroid in each dimension to be calculated in two cycles rather than 16. Thus, the entire motion detection and quantification process now takes 6 cycles on the PIPE, yielding an update rate of 10 HZ on calculating motion-energy centroids.

TRACKING THE OBJECT

The motion detection algorithm needs to translate motion in 2-D image space coordinates into motion in 3-D robot coordinates. We have restricted the motion of the object to a 2-D plane. Hence, we need to perform a simple calibration that will relate changes in image space coordinates to changes in the robotic workspace. In our experimental setup, the vertical distance from the camera to the moving object is fixed, so a simple calibration method is sufficient to relate changes in U-V image space with X-Y motion of the robotic arm.

Tracking the object is accomplished by having the PIPE system continually update the centroid measurement of the moving object A PC reads the centroid information from the PIPE upon a request from the robot arm when it is ready to perform another positioning move. The processing loop is:

- The system is initialized and when a moving object is detected by the spatio-temporal filters, a centroid of the motion energy is calculated and fed to the robot system.
- The robot is servoed so that the old centroid will map to the center of the camera's coordinate system, thus keeping the object centered in the field of view.
- The procedure is repeated as long as the spatio-temporal filters detect motion energy.

The connection between the image processing system and the robot arm is via a serial line running at 9600 Baud. This severely limits the ability of the system to do joint level servoing. The approach used here is to feed the new image centroid to the calibration transform on the arm system, and then move the arm to the new Cartesian coordinates. The arm controller has a number of parameters that can be used to tune this movement, including variable settling times and calculation of via points that smooth the response of the arm during its tracking phase. Currently, we can generate motion energy centroids at 10 HZ and arm servoing at 4 HZ. By removing the serial link, update rates of the arm position approaching 8 HZ may be possible.

EXPERIMENTAL RESULTS

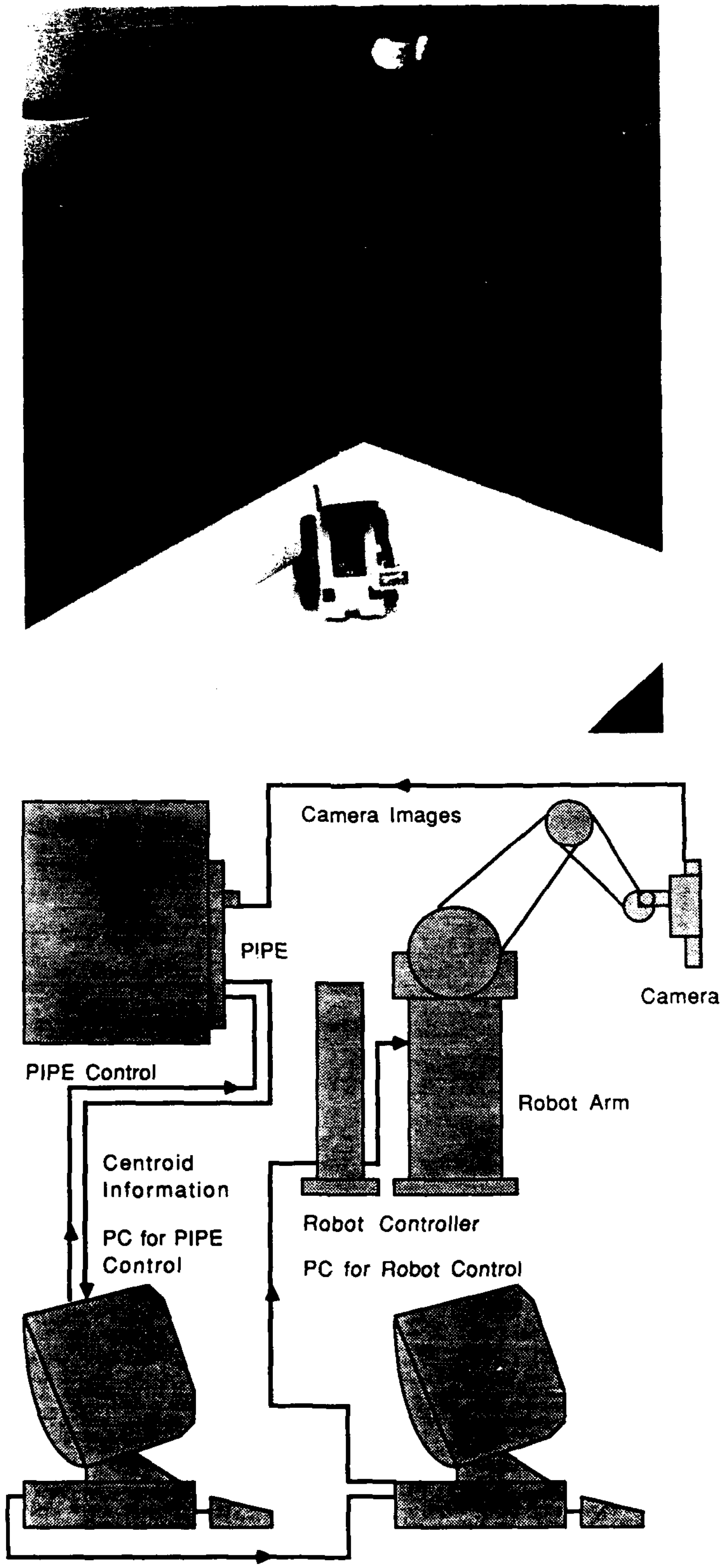
The system works quite well in motion tracking tasks. We have tested it in following straight line motion and in arbitrary randomized motion that is quite jerky and unpredictable, implemented with the mobile toy robot. The system is capable of tracking the object accurately as it covers the entire workspace of the robot. A video tape showing the tracking response also has been produced. Figure 3 shows 3 responses from the vision algorithm (from a static camera) for a typical tracking scene that shows the ability of the vision system to accurately locate motion energy using spatio-temporal filtering. In these images, the toy robot is moving inside a box, from which it bounces off randomly and changes direction. In the top image, the toy robot is moving in the upper left part of the scene, but it has encountered a corner of the containing box, which has slowed it to a stop. The motion energy is shown to the right, revealing no apparent motion (blank picture). In the middle image, the robot has started to bounce off the wall and motion energy is detected. Finally in the third image, the motion of the robot is tracked as it moves forward. The centroids of these detected motion energy regions are then used to guide the arm in its movement to track the object.

SUMMARY AND FUTURE WORK

Our future work is to include depth tracking as well as X-Y motion tracking. A possible approach is to use sonar [14] in the servo loop to add a depth value to the calibration. The update rates from the sonar are approximately equal to our current motion control rates, so the method seems compatible. In addition, recent work by Waxman, Wu and Bergholm [18] has exploited using multiple spatio-temporal filters on the PIPE to calculate image velocities, which we may be able to use to refine our motion velocity estimates.

ACKNOWLEDGEMENTS

This work was supported in part by Darpa contract N00039-84-C-0165, and the following NSF grants: DMC-86-05065, DCI-86-08845, CCR-86-12709, and IRI-86-57151. Support of the AT&T Foundation and Philips Laboratories is also gratefully acknowledged. Thanks to Randy Luck from ASPEX Inc., makers of the PIPE, for help on the PIPE implementations.



Figures 1 and 2: Arm mounted camera and mobile toy robot (top) and hardware configuration.

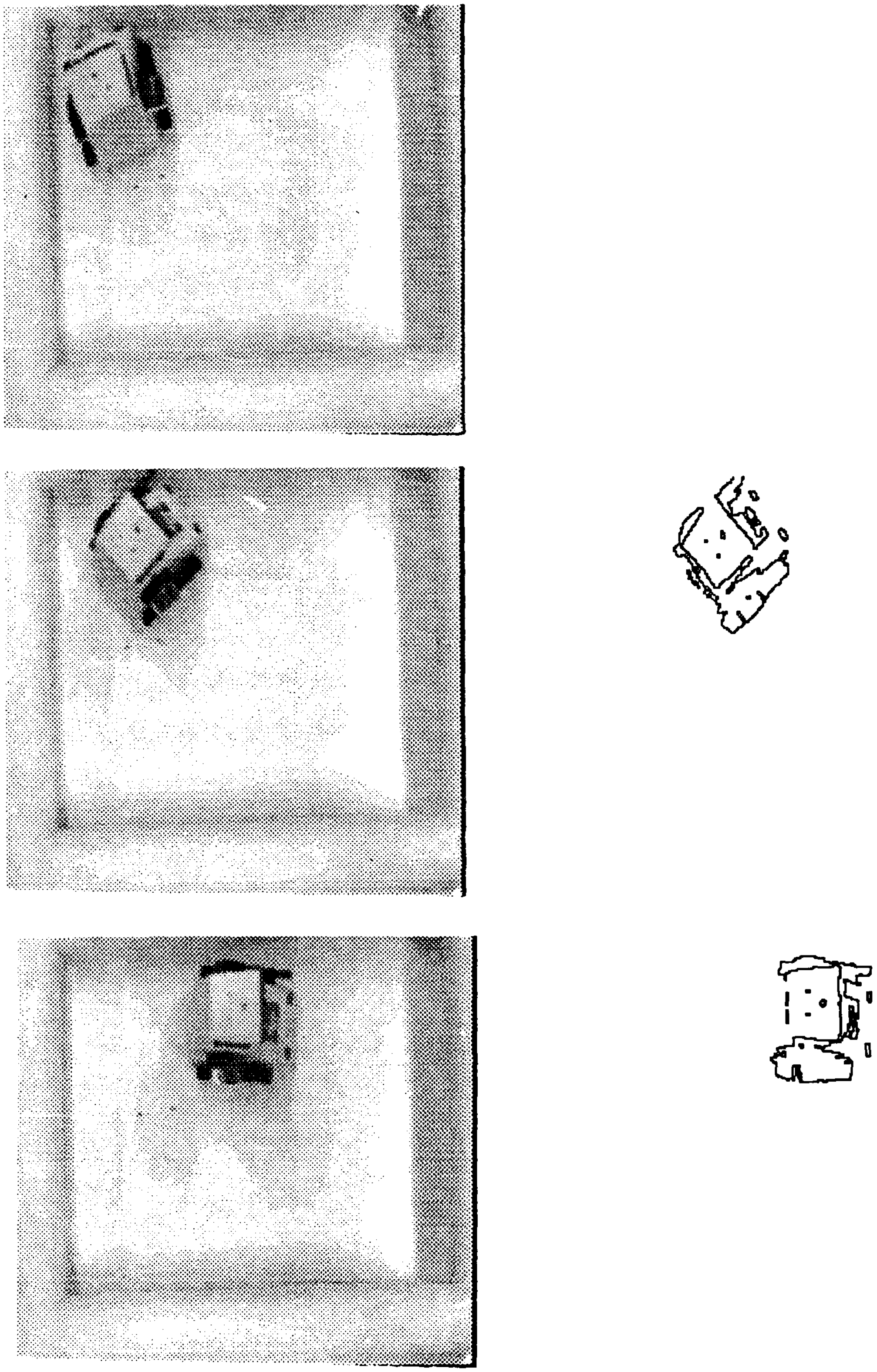


Figure 3: Sequence of 3 images (left) and associated motion energy (right).

References

1. Adelson, E. H. and J. R. Bergen, "Spatio-temporal enrgy models for the perception of motion," *Journal of the Optical Society of America*, vol. 2, no. 2, pp. 284-299, 1985.

2. Allen, Peter, Paul Michelman, and Kenneth S. Roberts, "An integrated system for dextrous manipulation," *IEEE Conference on Robotics and Automation*, Scottsdale, AZ, May 1989.

3. Anadan, P., *Measuring visual motion from image sequences*, Technical Report COINS-TR-87-21, COINS Department, University of Massachusetts, Amherst, 1987.

4. Aspex,, *PIPE User's Manual*, Aspex Incorporated, 530 Spring St., New York, NY.

5. Burt, P. J., J. R. Bergen, R. Hingorani, R. Kolczynski, W. A. Lee, A. Leung, J. Lubin, and H. Shvayster, "Object tracking with a moving camera," *IEEE Workshop on Visual Motion*, pp. 2-12, Irvine, CA, March 20-22, 1988.

6. Burt, P. J., C. Yen, and X. Xu, "Multi-resolution flow-through motion analysis," *Proceedings of the IEEE CVPR Conference*, pp. 246-252, 1983.

7. Buxton, B. F. and H. Buxton, "Computation of optic flow from the motion of edge features in image sequences," *Image and Vision Computing*, no. 2, 1984.

8. Corke, Peter, Richard Paul, and K. Wohn, *Video-rate visual servoing for sensory-based robotics*, Technical Report, Grasp Laboratory, Department of Computer and Information Science, University of Pennsylvania, Philadelphia.

9. Goldenberg, Richard, Wan Chi Lau, Alfred She, and Alan Waxman, "Progress on the prototype PIPE," *IEEE Conference on Robotics and Automation*, Raleigh, N. C., March 31 - April 3, 1987.

10. Heeger, David, "A model for extraction of image flow," *First International Conference of Computer Vision*, 1987.

11. Horn, B. K. P. and B. Schunk, "Determining optical flow," *Artificial Intelligence*, vol. 17, pp. 185-203, 1983.

12. Kent, E. W., M. O. Shneier, and R. Lumia, "PIPE: Pipelind image processing engine," *Journal of Parallel and Distributed Computing*, no. 2, pp. 50-78, 1985.

13. Lee, Sang Wook and K. Wohn, *Tracking moving objects by a mobile camera*, Technical Report MS-CIS-88-97, University of Pennsylvania, Department of Computer and Information Science, Philadelphia, November 1988.

14. Luo, Ren C., Robert E. Mullen Jr., and Daniel E. Wessell, "An adaptive robotic tracking system using optical flow," *IEEE Conference on Robotics and Automation*, pp. 568-573, Philadelphia, 1988.

15. Marr, David, *Vision*, W. Freeman, San Francisco, 1982.

16. Nagel, H. H., "On the estimation of dense displacement vector fields from image sequences," *Workshop on motion: Representation and Perception*, pp. 59-65, Toronto, 1983.

17. Scott, G. L., "Four-line method of locally estimating optic flow," *Image and Vision Computing, 5(2)*, 1986.

18. Waxman, Allen, Jian Wu, and Frederik Bergholm, "Convected activation profiles and the measurement of visual motion," *Proceedings CVPR 88*, pp. 717-723, Ann Arbor, MI, June 5-9, 1988.

19. Wiklund, Johan and Gosta Granlund, "Tracking of multiple moving objects," in *Time Varying Image Processing and Moving Object Recognition*, ed. V. Cappelini, pp. 241-249, Elsevier Science Publishers, 1987.

Dynamic Motion Vision

Joachim Heel
Artificial Intelligence Laboratory
Massachusetts Institute of Technology

Abstract

We present the idea of Dynamic Motion Vision in which a dynamical model of the motion imaging situation is the foundation for the systematic processing of a sequence of images. We use this formulation to design a Kalman Filter for the estimation of a dense depth map of the environment. We show further how a motion estimation procedure may be integrated into the filtering process. The resulting algorithm is a systematic solution for the problem of recovering both a dense depth map and motion parameters from an arbitrary sequence of images. Experimental results on synthetic and real images demonstrate the practicality of the approach.

1 Introduction

The key question in motion vision research today is how to extract information from an entire sequence of images rather than just two consecutive frames. The hope is that multiple frames will improve the accuracy, reduce the influence of noise and even allow the extraction of information which cannot be recovered from just two frames.

We can only hope for such an improvement if we succeed in incorporating the information from multiple frames in a systematic way derived from the physics of the motion imaging situation. Previously proposed solutions to this problem often lacked such a foundation or were subject to strong restrictions which limited their applicability.

Let us begin by examining what constitutes the temporal coherence in some of the currently proposed algorithms for the recovery of structure and motion. A first class of algorithms [2], [15], the rigidity scheme [16], uses intuitive or biologically motivated relationships between frames as the basis for multi-frame processing. Consequentially we have no guarantee for the quality of such an algorithm and failure cases are easily constructed.

Another group of algorithms makes use of multiple frames as a means of accumulating a sufficient number of constraints for the solution of a set of equations. Examples are [13], [12]. These methods do not make use of the temporal relationship between frames.

The photogrammetric problem of relative orientation motivated another approach: Find the motion parameters which minimize the spatial error of points in the real world corresponding to a given set of extracted features [7], [14]. Disadvantages of such approaches are the necessity to extract and match features and the complex nonlinear minimization problem involved

It was soon realized that only an explicit model of temporal behavior would allow further improvements. Dynamical models were proposed in [15], [3] and [4]. The initial approaches suffered from a number of problems which limited the practicality until Matthies, Szeliski and Kanade [10] added a new perspective recently. Using a simple dynamical model for temporal changes in depth they implemented a Kalman Filter based algorithm for the recovery of structure from real images with promising results. The remaining limitations are the necessity to know the camera motion as well as the restriction of motion to one translational degree of freedom.

In this paper we present two results: the application of the theory of dynamical systems to motion vision to establish an explicit model of the temporal relationship between image frames and an algorithm for the recovery of arbitrary structure and motion based on the dynamical model. We have conducted a series of experiments on synthetic and real images to verify the theoretical results.

The underlying paradigm of Dynamic Motion Vision bears applications for other tasks in motion vision such as segmentation, optical flow etc.

2 Dynamic Motion Vision

2.1 The idea

Dynamic Motion Vision is divided into two steps:

1. The formulation of a dynamical model for a quantity of interest.
2. The application of techniques developed for dynamical systems for the measurement of the particular quantity.

In our case the quantity of interest will be the environmental depth Z for which we construct a dynamical model relating it to the measurable optical flow. We apply to this model the technique of Kalman Filtering to obtain an estimate for the depth Z, i.e. the structure of the scene.

This formulation suggests that the approach bears applications far beyond the specific one presented here. We view it as a new perspective on the pertinent problems in motion vision. The theoretical foundation for this approach is described in [6].

2.2 A dynamical model for structure

A dynamical model[1] consists of two parts[2]:

1. A differential equation which describes the change of the quantity of interest $\mathbf{x}$ over time
$$\dot{\mathbf{x}} = \mathbf{f}(\mathbf{x}). \tag{1}$$
2. An algebraic equation which describes the relationship of the quantity of interest $\mathbf{x}$ to a measurable quantity $\mathbf{y}$
$$\mathbf{y} = \mathbf{g}(\mathbf{x}). \tag{2}$$

We now model the motion imaging situation in terms of such a dynamical system. We adopt the usual conventions for coordinate systems (origin at the center of projection, z-axis aligned with the optical axis pointing towards the image plane) and motion parameters (vectors of instantaneous translation $\mathbf{t}$ and rotation $\boldsymbol{\omega}$) as depicted in figure 1. Our quanitity of interest is the depth Z, we consider the optical flow (u, v) to be measurable.

If P is a point with position vector $\mathbf{R}$ on the object moving with $\mathbf{t}$ and $\boldsymbol{\omega}$ relative to the camera we have that
$$\dot{\mathbf{R}} = -\mathbf{t} - \boldsymbol{\omega} \times \mathbf{R}. \tag{3}$$
The third component of this vector differential equation
$$\dot{Z} = -W - (Ay - Bx)Z \tag{4}$$
is the dynamical model for the depth Z corresponding to (1). We have used the components of the motion vectors $\mathbf{t} = [U, V, W]^T$ and $\boldsymbol{\omega} = [A, B, C]^T$ and the coordinates (x, y) of the projection of P into the image plane.

[1] This is a simplified version of a general dynamical system.

[2] Boldface is used to denote vectors.

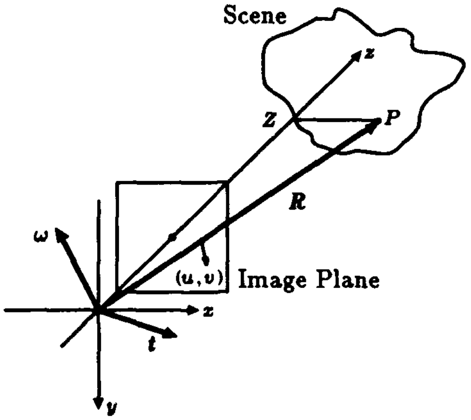


Figure 1: Coordinate system, motion parameters and projection geometry.

The relationship between our dynamic quantity Z and the measurable optical flow (u, v) is well known [9] as the motion field equations

$$\begin{aligned} u &= \frac{-U + xW}{Z} + Axy - B(x^2 + 1) + Cy \\ v &= \frac{-V + yW}{Z} - Bxy + A(y^2 + 1) - Cx. \end{aligned} \tag{5}$$

Figure 2 shows a block-diagram of the dynamical model we have constructed.

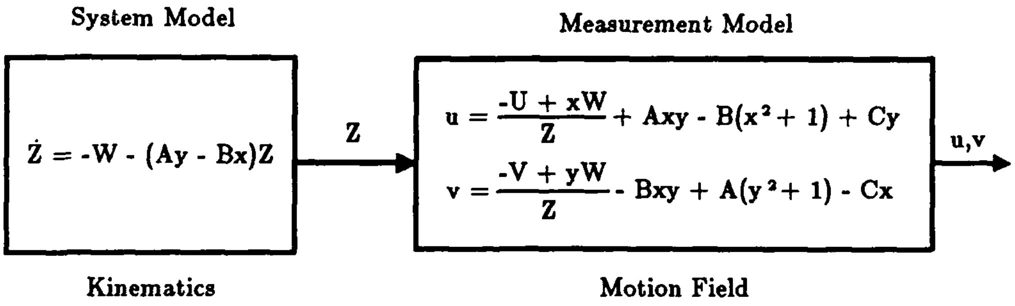

System Model
Measurement Model
$\dot{Z} = -W - (Ay - Bx)Z$
Z
$u = \frac{-U + xW}{Z} + Axy - B(x^2 + 1) + Cy$
$v = \frac{-V + yW}{Z} - Bxy + A(y^2 + 1) - Cx$
u,v
Kinematics
Motion Field

Figure 2: The dynamical systems model of motion vision

Clearly there are other dynamical models that can be constructed like the ones in [15], [3], [10]. The goal is to find the simplest one without making overly restrictive assumptions. Note that our only assumption at this point is the rigidity of the scene employed in (4).

Before we describe the algorithm which uses the above model for the recovery of structure and motion we briefly discuss the nature of the measurement process which provides us with the optical flow (u, v). We used the so-called sum-of-squared differences (SSD) optical flow. For every point $P_1 = (x_1, y_1)$ in image 1 we seek to determine the point $P_2 = (x_1 + u, y_1 + v)$ to which it moves in image 2[3]. We assume that the brightness in a neighborhood N of P_1 remains nearly unchanged between frames and that P_2 is within an area S of P_1. By denoting the brightness function in image i by $E_i(x, y)$ we can formalize the procedure. For every point (x, y) in image 1 we seek the displacement (u, v) such that

$$SSD(u,v) = \min_{u,v \epsilon S} \sum_{\xi,\eta \epsilon N} [E_1(x+\xi, y+\eta) - E_2(x+\xi+u, y+\eta+v)]^2. \tag{6}$$

In other words we are looking for the displacement that minimizes the sum of squared differences. Various improvements of this basic technique using multi-grid techniques and sub-pixel fitting are possible.

For our purposes we are mainly interested in quantifying the certainty or variance of an optical flow value obtained by this method. We will make use of the variances σ_u^2 and σ_v^2 of the optical flow measurement in our structure and motion algorithm. Anandan [1] suggested a "confidence measure" which intuitively corresponds to an inverse variance in some way. In Heel [6] we give a formal derivation of optical flow variance based on noise propagation.

Under some assumptions these variances take the form

$$\sigma_u^2 = \frac{SSD}{SSD_{xx}^2} \quad \text{and} \quad \sigma_v^2 = \frac{SSD}{SSD_{yy}^2} \tag{7}$$

where the subscripts denote partial differentiation and expressions are evaluated at the (u, v) that minimize (6). Intuitively we see that the variances are high where the SSD is large or in other words the match between the two compared regions was not very good. The variances are also high in the case of small second derivatives which indicate a low curvature of the SSD surface. Such a case arises when the region we are searching for a match is nearly uniform in brightness so that many matches are possible.

2.3 A Kalman Filter for depth estimation

The problem we are facing is to compute a dynamically changing quantity (in our case the depth Z) but we are able to measure only a function of this quantity (the optical flow). However, we have a mathematical model which describes the temporal evolution of the quantity of interest. Given an initial value, the depth Z will behave exactly according to this dynamical model and we could determine Z at any time precisely. The problem is that we do not know this initial value.

To solve this problem we will simulate the differential equation describing the dynamical system in our computer in parallel with the actual system (the moving camera) starting with our best guess for the initial value. In every step we can then compare the optical flow values produced by our simulation with the one we measure from our image. To compensate for the error we made in our choice of the initial value we then use the difference between the measured and simulated optical flow to improve our simulation value for the depth. This is the well-known idea of an *observer* depicted in figure 3.

The dynamics simulation part of the observer is exactly the same as the dynamic model (4) we constructed in the previous section, however, we use a caret to denote quantities in the observer.

$$\dot{\hat{Z}} = -W - (Ay_k - Bx_k)\hat{Z} \tag{8}$$

Following the observer principle we improve the simulation of Z in every step by an amount proportional to the error in the observed quantity, the optical flow:

$$\hat{Z} = \hat{Z} + \mathbf{e}^T [\begin{bmatrix} u \\ v \end{bmatrix} - \begin{bmatrix} \hat{u} \\ \hat{v} \end{bmatrix}] \tag{9}$$

[3] Velocities are normalized by the interframe time interval T to obtain displacements.

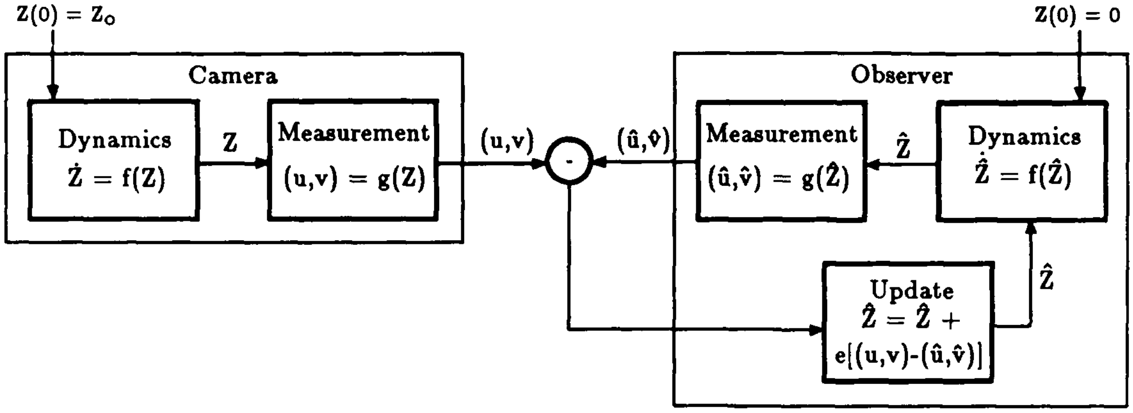


Figure 3: The observer principle

where $(\hat{u}, \hat{v})$ is computed from (5) using the current value of $\hat{Z}$.

The obvious question is how to pick the elements of the so-called gain e. Here is where the reliability of our measurements expressed in the form of variances σ_u^2 and σ_v^2 comes into play. Clearly, if we consider our measurements (u, v) to be very reliable (small variance) then we would like the error in the measurement to have a very strong effect on the value of Z. On the other hand a low confidence in the measurement should cause us to weight an error in the optical flow less. A systematic way of computing the gain e which takes the variance of the measurements into account is the Kalman Filter which interprets both the state Z and the measurement (u, v) as stochastic processes. The gain is chosen such that the signal-to-noise ratio of the estimation is minimized.

For details on the Kalman Filter, specifically the nonlinear version used here, refer to [5]. We summarize below the discrete time filter equations for the recovery of structure from optical flow.

1. *Update component:*

   Filter Gain $\quad \mathbf{e}_k^T = \sigma_{Z,k}^2 \mathbf{c}_k^T [\mathbf{c}_k \mathbf{c}_k^T \sigma_{Z,k}^2 + \mathbf{R}_k]^{-1}$

   State Update $\quad \hat{Z}_k = \hat{Z}_k + \mathbf{e}_k^T [\begin{bmatrix} u_k \\ v_k \end{bmatrix} - \begin{bmatrix} \hat{u}_k \\ \hat{v}_k \end{bmatrix}]$

   Covariance Update $\quad \sigma_{Z,k}^2 = (1 - \mathbf{e}_k^T \mathbf{c}_k) \sigma_{Z,k}^2$

2. *Prediction component:*

   State Prediction $\quad \hat{Z}_{k+1} = -TW + [1 - T(Ay_k - Bx_k)]\hat{Z}_k$

   Covariance Prediction $\quad \sigma_{Z,k+1}^2 = [1 - T(Ay_k - Bx_k)]^2 \sigma_{Z,k}^2$

We have used the following notation: $\hat{Z}_k$ and $\sigma_{Z,k}^2$ are the depth and its variance, the quantities the filter estimates. Further we have

$$\mathbf{c}_k = \begin{bmatrix} (U - x_k W)/Z_k^2 \\ (V - y_k W)/Z_k^2 \end{bmatrix} \tag{10}$$

and

$$\mathbf{R}_k = \begin{bmatrix} \sigma_u^2 & 0 \\ 0 & \sigma_v^2 \end{bmatrix}. \tag{11}$$

the measurement covariance matrix. Note that we have assumed that the estimates of the optical flow components u and v are independent. Finally, $\hat{u}$ and $\hat{v}$ ir the state update equation denote the motion field computed according to (5).

2.4 Making the algorithm practical

Let us briefly reflect upon the effect of the technique outlined above. A point P in the environment will appear at $P' = (x, y)$ in the image plane. The state equation of our system describes, how the depth of P will change due to the motion $\mathbf{t}$, $\boldsymbol{\omega}$. The Kalman Filter uses this knowledge to reconstruct the depth Z from the optical flow measurement (u, v). Note that the filter produces only the result for Z at P' or in other words: we need one Kalman Filter at every pixel in order to obtain a dense depth map.

2.4.1 Depth reinterpolation

The previous remark uncovers the main difficulty we are faced with in implementing the technique: The point P' at which the Kalman Filter operates also moves inbetween frames. In practice, this will have the following consequence. Suppose the current depth estimate is stored in an array corresponding to the image pixel array. The updated value for Z computed at (x, y) is not valid at (x, y) but rather at some (x', y') which is the location to which (x, y) moves due to the relative motion of the scene. Fortunately, (x', y') is given to us by the optical flow vector (u, v) which would allow us to have the filter "follow" the projection real world point in the image plane.

Instead of pursuing this option which may be realistic for a small set of features but hardly for a dense depth map we insert a reinterpolation stage which computes the Z values at the grid points of our depth map from the warped depth map. A number of techniques are available for this task including bicubic and bilinear interpolation. We have found a fast weighted averaging scheme described in [6] to work very well, while more sophisticated methods are computationally expensive.

2.4.2 Motion estimation

So far we have acted as if the motion parameters $\mathbf{t}$ and $\boldsymbol{\omega}$ were known. In practice this may be the case, for instance in a mobile robot system where the vehicle motion can be extracted from axle potentiometers or similar sensors. However, the iterative nature of the Kalman Filter allows us to estimate the motion parameters in addition to the structure. Note that the scheme will work even better if the parameters are in fact known.

The idea for the motion estimation is based on the fact that if the structure Z were known to us we could use it together with at least 3 measurements of the optical flow (u, v) to compute $\mathbf{t}$ and $\boldsymbol{\omega}$ in a least-squares fashion from the motion field equations (5)

$$u = \frac{-U + xW}{Z} + Axy - B(x^2 + 1) + Cy \tag{12}$$

$$v = \frac{-V + yW}{Z} - Bxy + A(y^2 + 1) - Cx. \tag{13}$$

The key idea is that, since Z is not known (it is precisely what the filter is estimating) we use the current estimate $\hat{Z}$ instead. For the least-squares estimation, let us assume the we have $i = 1, \ldots, n$ locations (x_i, y_i) in the image where we measured the optical flow (u_i, v_i) and estimated the depth estimates Z_i (the output of the prediction stage of the filter). Then we wish to find $\mathbf{t}$ and $\boldsymbol{\omega}$ such that

$$\begin{aligned}\Theta &= \sum_{i=1}^{n} (u_i + [\frac{1}{Z_i}, 0, -\frac{x_i}{Z_i}] \cdot \mathbf{t} + [x_i y_i, -(x_i^2+1), y_i] \cdot \boldsymbol{\omega})^2 + \\ &\qquad (v_i + [0, \frac{1}{Z_i}, -\frac{y_i}{Z_i}] \cdot \mathbf{t} + [y_i^2 + 1, -x_i y_i, -x_i] \cdot \boldsymbol{\omega})^2 \qquad (14) \\ &= \sum_{i=1}^{n} (u_i + \mathbf{a}_i \cdot \mathbf{t} + \mathbf{b}_i \cdot \boldsymbol{\omega})^2 + (v_i + \mathbf{c}_i \cdot \mathbf{t} + \mathbf{d}_i \cdot \boldsymbol{\omega})^2\end{aligned}$$

is minimal. Differentiating with respect to the motion parameters **t** and ω yields the following linear system of 6 equations as a necessary condition for a minimum

$$\begin{aligned}\left[\sum_{i=1}^{n}(\mathbf{a}_i^2 + \mathbf{c}_i^2)\right]\mathbf{t} + \left[\sum_{i=1}^{n}(\mathbf{a}_i \cdot \mathbf{b}_i + \mathbf{c}_i \cdot \mathbf{d}_i)\right]\boldsymbol{\omega} &= -\sum_{i=1}^{n}(u_i\mathbf{a}_i + v_i\mathbf{c}_i) \\ \left[\sum_{i=1}^{n}(\mathbf{a}_i \cdot \mathbf{b}_i + \mathbf{c}_i \cdot \mathbf{d}_i)\right]\mathbf{t} + \left[\sum_{i=1}^{n}(\mathbf{b}_i^2 + \mathbf{d}_i^2)\right]\boldsymbol{\omega} &= -\sum_{i=1}^{n}(u_i\mathbf{b}_i + v_i\mathbf{d}_i) \qquad (15)\end{aligned}$$

This system can easily be solved for the desired parameters.

The result of integrating the depth reinterpolation and the motion estimation into the block diagram of the Kalman Filter is shown in figure 4.

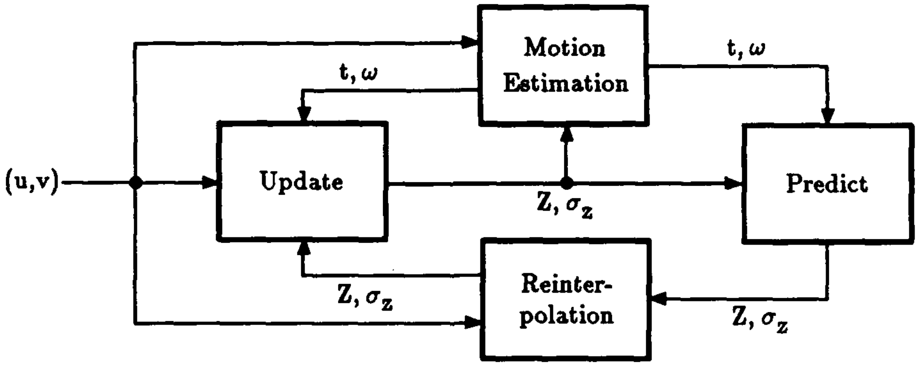


Figure 4: The block diagram of the Dynamic Motion Vision algorithm for depth and motion estimation.

2.5 Implementation and Experiments

We have implemented the Dynamic Motion Vision structure and motion estimation from optical flow and present below the results of experiments on synthetic and real images. Since space is limited we present only the most important results for the synthetic image experiment and report the results on real images in full detail.

In the first experiment we used a 3D polygonal modeler and computer graphics package to generate a sequence of images of a crater as perceived by an aircraft flying over the structure. Some images from the sequence, samples of the optical flow computed from them and the recovered structure are shown. This experiment allows us to compare the structure used in the creation of the images with the depth map recovered by the algorithm.

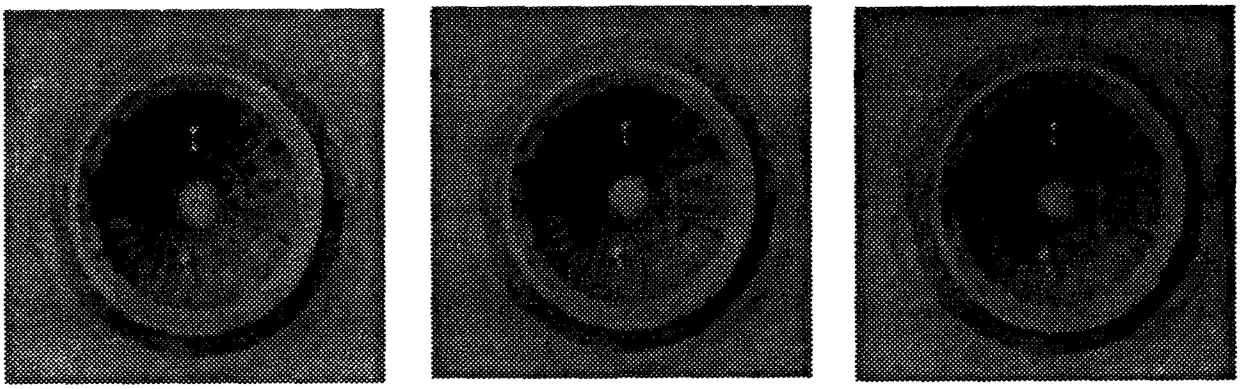


Figure 5: The first three images of the crater sequence.

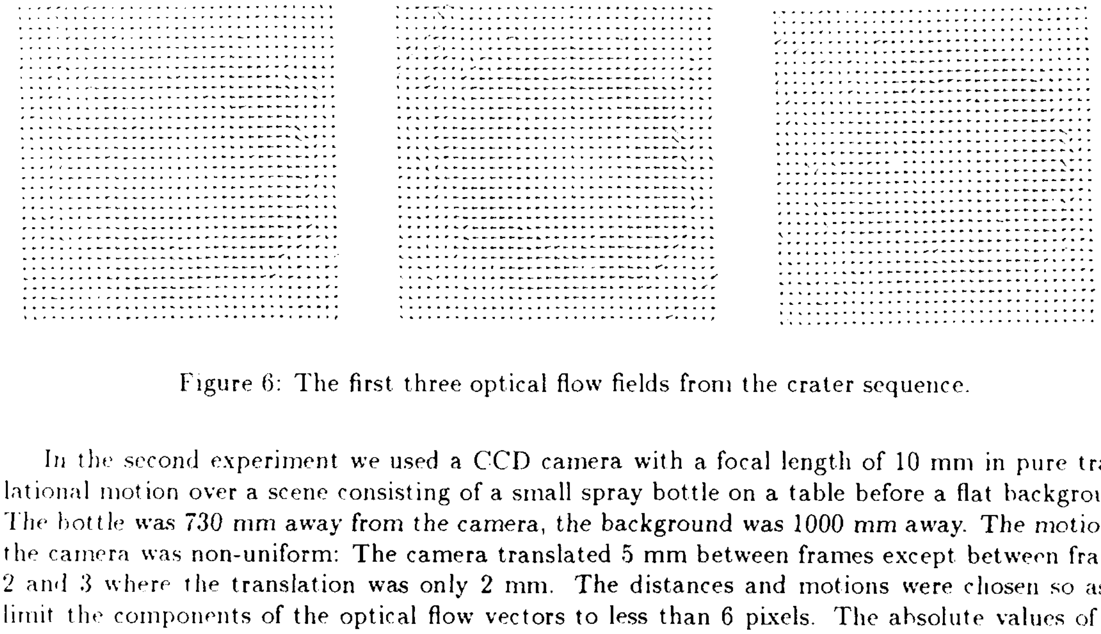


Figure 6: The first three optical flow fields from the crater sequence.

In the second experiment we used a CCD camera with a focal length of 10 mm in pure translational motion over a scene consisting of a small spray bottle on a table before a flat background. The bottle was 730 mm away from the camera, the background was 1000 mm away. The motion of the camera was non-uniform: The camera translated 5 mm between frames except between frames 2 and 3 where the translation was only 2 mm. The distances and motions were chosen so as to limit the components of the optical flow vectors to less than 6 pixels. The absolute values of the translational motion parameters and depth are of course only recovered up to a constant factor which is determined by the choice of the initial value of Z. In all experiments we began with an initially flat depth map the value of which was an average of the actual depth in the scene as it may be obtained by a simple range sensor.

We considered this experiment to be a crucial test for the algorithm for the following reasons.

1. The motion was nonuniform and therefore bound to reveal any problems due to the simultaneous estimation of depth and motion.

2. The depth variations in the scene are small. As a consequence the difference between the optical flow on the bottle and on the background would be very small.

3. Due to absence of texture on the bottle the optical flow there was rather noisy.

The algorithm seems to handle all of these problems reasonably well. Since no assumption about the motion was employed, the non-uniformity was detected as the plot 10 of the motion parameters

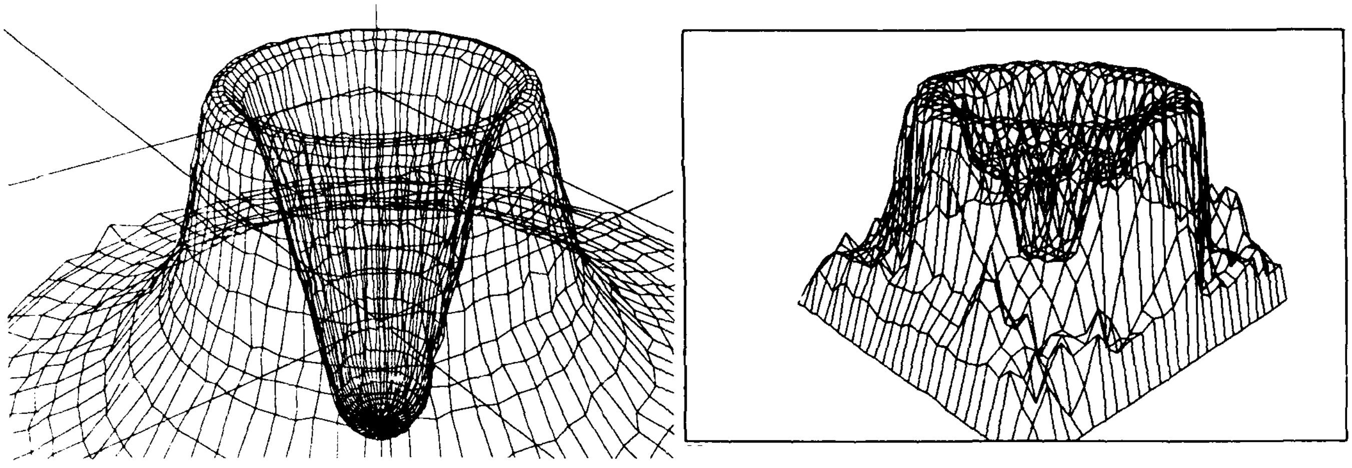


Figure 7: The structure created in the geometric modeler and the recovered structure. Note that viewing perspectives differ slightly.

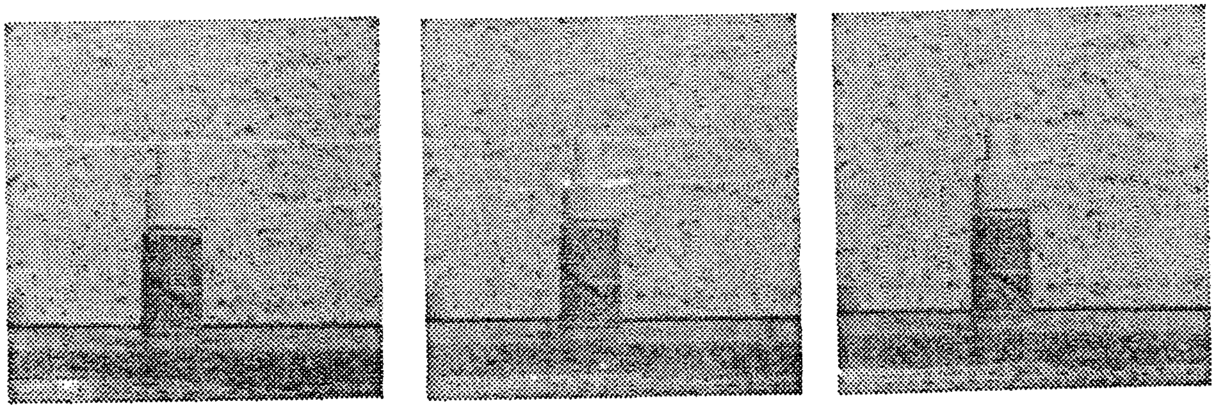


Figure 8: The first three images of the bottle sequence.

reveals. We observed that the z-component of the translation vector is particularly sensitive to errors in the optical flow/depth. The structure of the bottle, the table is on which it stands and the flat background are clearly recovered. As expected the scarcely textured bottle cap's structure is not recovered as accurately.

The implementation is also reasonably fast: for the image size of 300 by 300 pixels one iteration takes approximately 2 minutes on a Sun 3/60 if the optical flow has been precomputed. Note, however, that due to the local nature of the Kalman Filters this algorithm is ideal for a parallel implementation on a SIMD machine and a considerable speedup should be achievable.

Our other experiments were equally encouraging with exception perhaps of those involving a focus of expansion in the image plane. Near the focus of expansion, the nature of the motion forces the optical flow measurements to very small values thereby decreasing the signal-to-noise ratio. A look at the equations reveals that the Kalman Filter reacts accordingly by choosing very small values for the filter gain. This is of course the desired behavior but the depth recovered in a region around the FOE becomes useless.

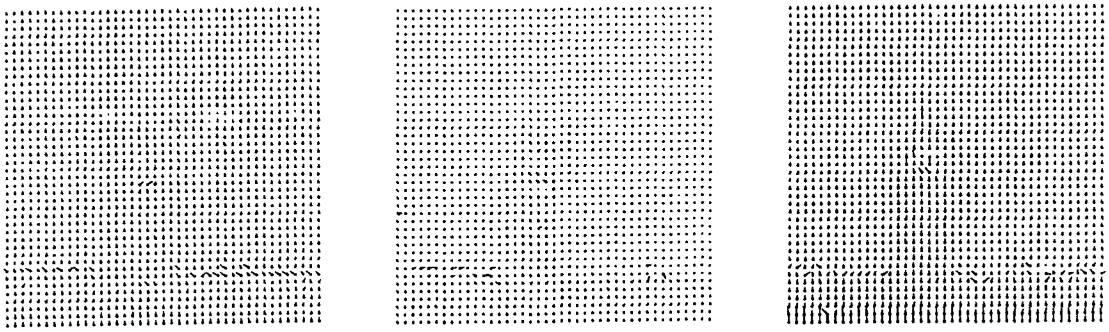


Figure 9: The first three optical flow fields from the bottle sequence.

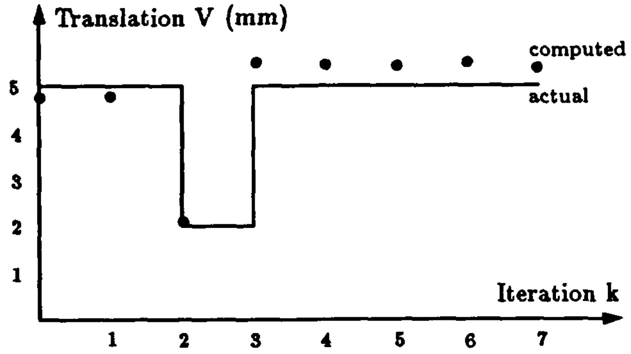


Figure 10: The recovered and the actual translational motion. All other motion components were negligibly small with exception of W which had a peak error of -0.12 mm in iteration 3.

3 Conclusion

We have shown how the theory of dynamical systems can be applied to the motion imaging situation. The resulting formulation can be exploited in the design of a Kalman Filter for the estimation of a dense depth map with interleaved motion estimation. We have also shown, that Dynamic Motion Vision need not be be subject to the severe limitations of previous approaches. We need no knowledge of motion or features. Note however, that the approach will work particularly well in the case of known motion or at location of feature-points.

A major shortcoming of the specific application of Dynamic Motion Vision outlined in this paper is the use of optical flow as a measurement. Optical flow computation is not only computationally expensive but also very noisy in general. Our goal is to formulate the motion imaging situation in terms of a dynamical system in which the brightness values or their derivatives appear directly as measurements. We will build on the work of Horn, Weldon and Negahdaripour [11], [8] who have developed *direct* methods for two-frame motion vision.

We believe that Dynamic Motion Vision can provide solutions to other problems than Kalman Filters for structure recovery. In particular we intend to use dynamical models for the segmentation of scenes containing several moving objects and for the improvement of structure estimates obtained by a stereo pair of cameras in motion. In our opinion, the most intriguing feature of Dynamic Motion

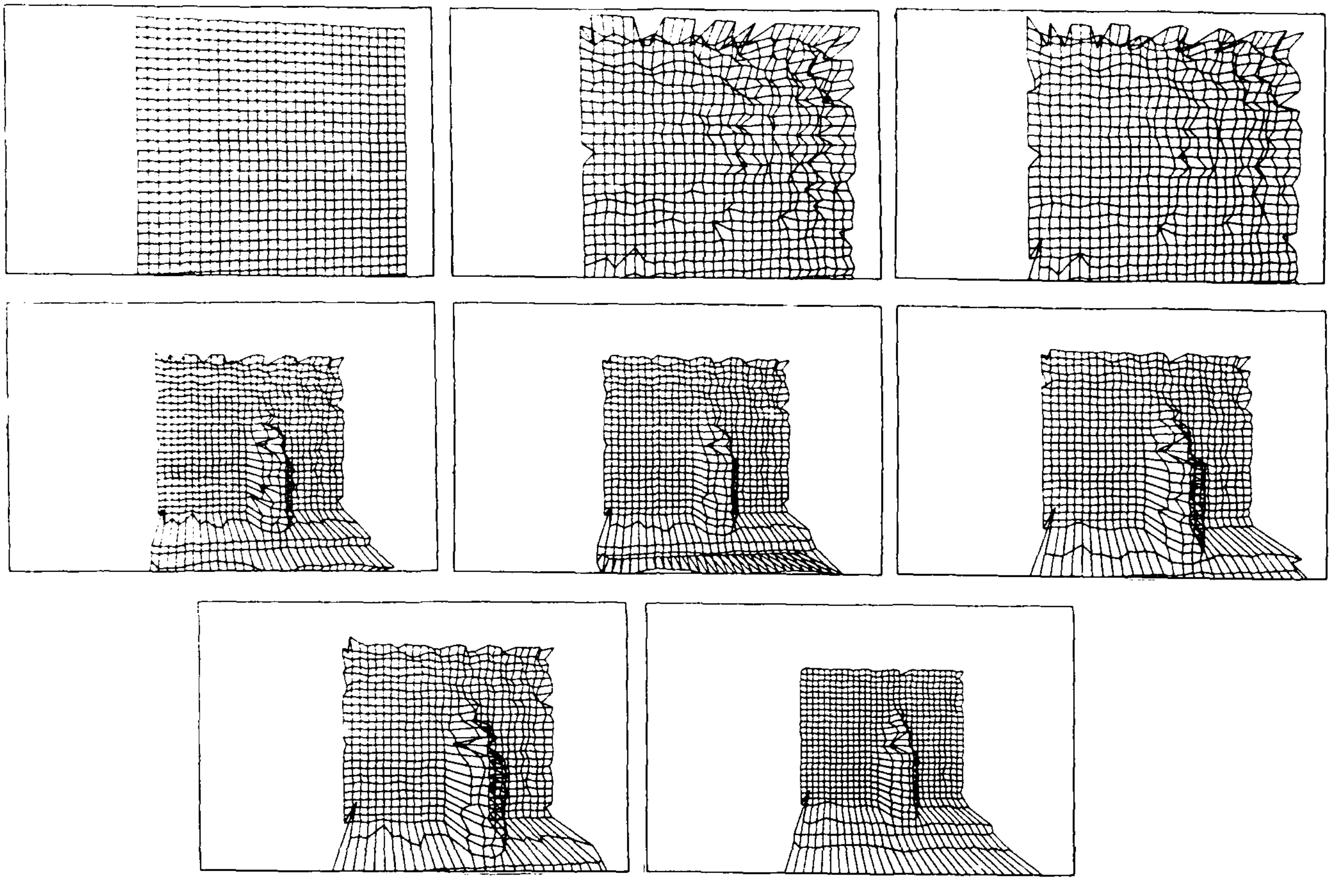


Figure 11: The structure recovered after each iteration step of the DMV algorithm from left to right and top to bottom.

Vision is that it provides a systematic way of processing a sequence of images and is no longer limited by the restrictions of previous techniques.

4 Acknowledgements

This report describes research done at the Artificial Intelligence Laboratory of the Massachusetts Institute of Technology and Siemens Research and Technology Laboratory, Princeton. Support for the Artificial Intelligence Laboratory's research is provided in part by the Advanced Research Projects Agency of the Department of Defense under Office of Naval Research contract N00014-85-K-0124.

Many thanks to Berthold Horn for his comments on the ideas in this paper.

References

[1] P. Anandan. *Computing dense displacement fields with confidence measures in scenes containing occlusion*. COINS Technical Report 84-32, University of Massachusetts, Amherst, December 1984.

[2] S. Bharwani, E. Riseman, and A. Hanson. Refinement of environment depth maps over multiple frames. In *Proceedings of the Workshop on Motion: Representation and Analysis*, Charleston, S. C., May 1986.

[3] T. J. Broida and R. Chellappa. Estimation of object motion parameters from noisy images. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, PAMI-8(1), January 1986.

[4] T. J. Broida and R. Chellappa. Kinematics and structure of a rigid object from a sequence of noisy images. In *Proceedings of the IEEE Workshop on Motion: Representation and Analysis*, Charleston, SC, May 1986.

[5] A. Gelb (Ed.). *Applied Optimal Estimation.* The MIT Press. Cambridge, MA, 1974.

[6] J. Heel. Dynamical systems and motion vision. AI Memo 1037, MIT Artificial Intelligence Laboratory, April 1988.

[7] B. K. P. Horn. Relative orientation. AI Memo 994, MIT Artificial Intelligence Laboratory, September 1987.

[8] B. K. P. Horn and E. J. Weldon, Jr. Direct methods for recovering motion. *International Journal of Computer Vision*, 2, 1988.

[9] H. C. Longuet-Higgins and K. Prazdny. The interpretation of a moving retinal image. In S. Ullman and W. Richards, editors, *Image Understanding 1984*. Ablex, 1984.

[10] L. Matthies, R. Szeliski, and R. Kanade. Kalman filter-based algorithms for estimating depth from image sequences. In *Proceedings of the DARPA Image Understanding Workshop*, Cambridge, MA, April 1988.

[11] S. Negahdaripour and B. K. P. Horn. Determining 3-d motion of planar objects from brightness patterns. In *Proceedings of Ninth International Joint Conference on Artificial Intelligence*, 1985.

[12] J. W. Roach and J. K. Aggarwal. Determining the movement of objects from sequences of images. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, PAMI-2(6), November 1980.

[13] H. Shariat and K. E. Price. How to use more than two frames to estimate motion. In *Proceedings of the Workshop on Motion: Representation and Analysis*, Charleston, S. C., May 1986.

[14] M. Spetsakis and J. Aloimonos. A multi-frame approach to visual motion perception. Technical Report CAR-TR-407, Computer Vision Laboratory, Center for Automation Research, University of Maryland, November 1988.

[15] J. Stuller and G. Krishnamurthy. Kalman filter formulation of low-level television motion estimation. *Computer Vision, Graphics and Image Processing*, 21(2), February 1983.

[16] S. Ullman. Maximizing rigidity: The incremental recovery of 3-d structure from rigid and rubbery motion. AI Memo 721, MIT Artificial Intelligence Laboratory, June 1983.

A Data Set for Quantitative Motion Analysis

R. Dutta R. Manmatha L. R. Williams E. M. Riseman

Computer and Information Science Dept.
University of Massachusetts at Amherst
Amherst, MA 01003

Abstract

This paper [1] discusses the collection of image sequences for quantitative experiments in motion analysis. Much of the work in motion analysis aims at the determination of motion parameters and depth of environmental points. In order to assess the effectiveness of motion algorithms it is necessary to obtain motion data with ground truth of known accuracy. Until now, such accurate ground truth has not been available, especially in the context of autonomous navigation in outdoor environments. Availability of good ground truth is critically important in understanding the limits and accuracy of motion analysis, for without such data collected under realistic conditions, the effectiveness of such algorithms is a matter of conjecture. This paper contains a description of the motion data collected by the Computer Vision Group from the Computer and Information Science Department, University of Massachusetts. The data was collected at Martin Marietta in Denver in October 1988, using the imaging facilities provided by the Autonomous Land Vehicle (ALV). A total of eight sequences of about 30 frames each were collected at five different outdoor sites using *move and shoot* and *stop and shoot* methods. For all the sequences accurate ground truth of environmental objects was determined using theodolites and a laser range finder. The camera egomotion parameters (this included both translation and rotation) were determined using a *Land Navigation System* on board the ALV. We show example images of the data set collected and present an analysis of one of the sequences. The data will be made available for motion research to the scientific community.

1 Introduction

Much of the motion analysis effort is aimed at the determination of motion parameters and depth of environmental points. In order to assess the effectiveness of motion algorithms it is necessary to obtain motion data with ground truth of known accuracy. Until now, such accurate ground truth has not been available, especially in the context of autonomous navigation in outdoor environments. Availability of good ground truth is critically important in understanding the limits and accuracy of motion analysis, for without data collected under conditions, the effectiveness of such algorithms is a matter of conjecture.

It should be noted that various studies [1,3,4,5,6,7,8] demonstrate that algorithmic computation of motion parameters and depth are subject to error in many circumstances, especially those situations associated with realistic applications. To our knowledoe, accurate ground truth for motion imagery of outdoor scenes does not exist.

The problem is compounded by the fact that some algorithms are very sensitive to small variations in the data. For example it has been shown [7] that algorithms assuming purely translational motion are very sensitive to small amounts of rotation; this rotation can be caused by uneven surface grade, bumps, stones, road banking, or vehicle sway and vibration. Since such deviations from ideal trajectories will be present in any real application, they must be taken into account by motion algorithms. Thus, to evaluate motion algorithms it is important to obtain accurate measurements of the actual motion of the vehicle, as well as the depths of the objects in the scene.

Since accurate ground truth has not been available our first goal was to collect image sequence data along with accurate ground truth information. Our previous experience with relatively small scale data acquisition with the NAVLAB at Carnegie-Mellon and the University of Massachusetts mobile robot encouraged us to undertake this venture. For this paper we used the facilities provided by the Autonomous Land Vehicle (ALV) at Martin Marietta. It is our expectation that the images and the associated data will be useful to researchers for both qualitative and

[1] This research was supported by Defense Advanced Research Projects Agency grants F30602-87-C-0140,DACA76-85-C-0008, DACA76-86-C-0015 and National Science Foundation grant DCR8500332.

quantitive motion work.

The main body of this paper discusses the approach we have adopted to determine accurate ground truth and gives the experimental results. This include sample images and an analysis of one of the image sequences. After the experimental results we state some of the intended uses of the data.

2 Experimental Methodology

Figure 1: **Photograph of the ALV**

Collection of motion data is itself a non-trivial task. The motion data set needs to consist of more than just a series of images acquired at intervals. First, considerable instrumentation of the vehicle and camera is required in order to provide the camera position and look angle with respect to a 3-D world coordinate system for each image of the set. In addition to this accurate ground truth, positions of a number of environmental objects are needed in order to assess the accuracy of the results. We identified two different methods for collecting motion data. The first is a *stop and shoot* method of data collection, while the second is a *move and shoot* method. In the *stop and shoot* method of data collection the vehicle is stopped while acquiring each image frame. After an image frame has been acquired the vehicle is moved again and another image frame acquired. This goes on until all the frames have been collected. The ALV's Land Navigation System (LNS) is used to provide accurate vehicle position and pointing information. The position and pointing parameters of the camera with respect to the vehicle needs to be determined by prior calibration. Accurate 3-D positions of a number of environmental objects are determined from a detailed survey of the area; for cultural (i.e. manmade) objects like buildings, fences and signposts, the ground truth will be fairly accurate. For some natural objects like rocks, it will be possible to determine the position of specific surface points accurately, while for other natural objects like bushes and trees it is more difficult to determine a single location representative of their position. In some cases additional objects like traffic cones were added to provide more detail. These were precisely located by surveying the point on top of the cone. Note that stop and shoot may not reflect the perturbations of the vehicle that will probably be encountered by real moving vehicles, nor will it reflect the accuracy with which the LNS can dynamically acquire data. A second problem of course is the fact that light levels may often change due to passing clouds or other factors.

In the *move and shoot* method of data collection the same measurements are made using the same instrumentation, but the vehicle is kept in motion. Important advantages of *move and shoot* are the realistic scenario of a moving vehicle and the ability to shoot images rapidly before variations in lighting become significant. The motion of the vehicle will introduce certain errors. First, certain data (such as the vehicle roll angle) are available only at 40

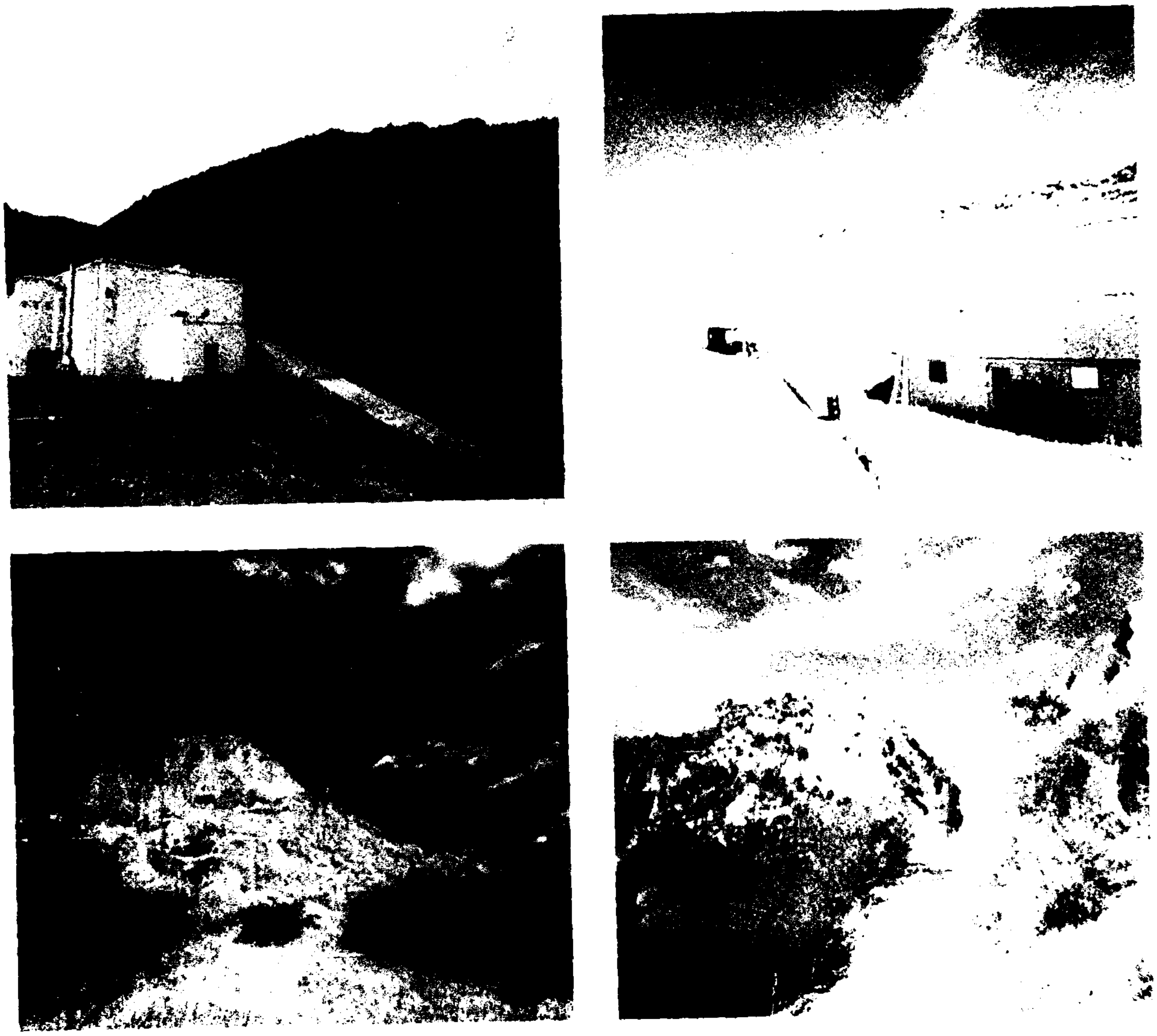
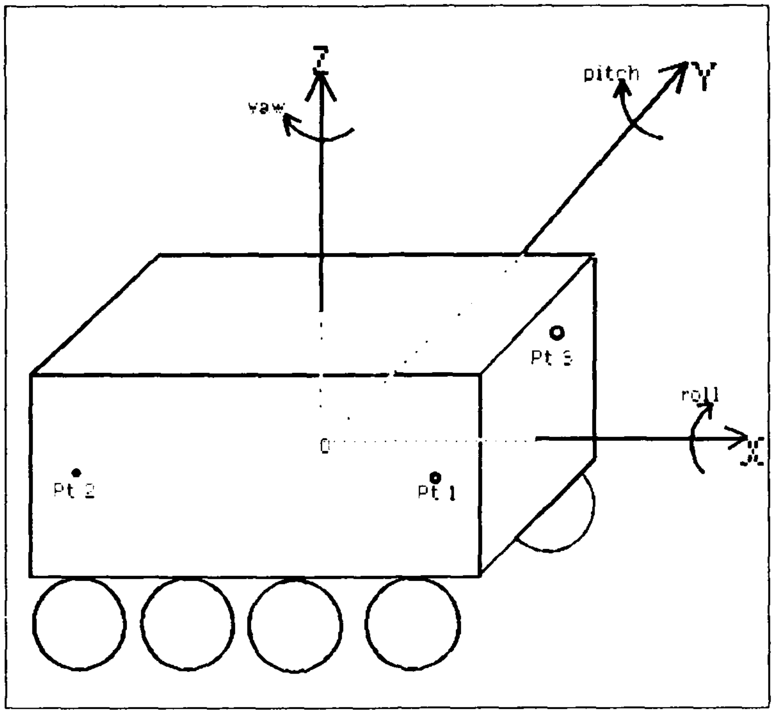


Figure 2: **Site 2 (top left), Site 3(top right), Site 4(bottom left), Site5(bottom right)**

Table 1: **Description of Images Acquired. Most image sequences are 30 frames long.**

Site	Name	Description	Surface	Methodology
1	Rocket Field	hill, cones, bldgs.	dirt-road	stop & shoot
2	Record Bldg.	road, bldg., hill	on-road	stop & shoot
2	"	"	"	move & shoot
3	Rifle Range	road, sheds, bldgs.	dirt-road	st · & shoot
3	"	"	"	move & shoot
3	"	"	"	move & shoot
4	Beyond Tomorrow	giant rock, trees	off-road	move & shoot
5	Stage Coach Road	hills, rocks, trees	dirt-road	move & shoot

Figure 3: **Coordinate System of the LNS**



ms intervals. Second, slight timing mismatches between measurements and image acquisition will introduce errors in the data set. This can occur if the vehicle hits a bump between the reading of the LNS and the acquisition of the image. Finally, it is assumed that the camera is fixed rigidly to the vehicle. No account is taken of movement of the camera relative to the vehicle attributable to vibration or deformation of the shell, and no account is taken of slight movements of the camera optics from vibrations or movements. Another problem with this approach arises from the fact that image acquistion and digitization may take substantial time because of the limited in memory storage and the consequent need to move large amounts of data before re-shooting.

To provide truly representative data, various combinations of sites and data collection methods were considered. Thus the kind of road surface considered *(on-road,dirt-road and off-road)* may determine the practicality of certain kinds of motion algorithms (e.g. algorithms which assume smooth motion between frames will work best with on-road sequences and may not work at all with off-road sequences).

The sequences acquired also differ in scene content. Some road sequences have many man-made objects (such as buildings and fences), since algorithms based on lines are well equipped to exploit these. Finally, to provide a complete data set, there are sequences in which there are no man-made objects but instead only natural objects like rocks, trees and bushes. Vehicles must be able to navigate in these environments as well. Most of the sites have a number of distinct objects or environmental features, but are not overly cluttered, since it was deemed infeasible to determine the location of every object visible at such a site. In a few of the sequences a small number of traffic cones and cardboard boxes were placed throughout the scene. Survey data was used to locate the most distinguishable objects.

Image and instrumentation data was collected for sequences of approximately 30 vehicle locations using the *stop and shoot* method. The image and instrumentation data was collected again using the *move and shoot* method. For both methods survey data was used to locate the initial and final positions of the vehicle (so that it could be related to the survey data for the objects). Additionally for one of the *stop and shoot* sequences, the vehicle's position was surveyed at every point to provide an independent check of the LNS data. In this context it should be mentioned that the ALV (Fig 1) is 2.7 m wide, 4.2 m long and 3.1 m high. It weighs approximately 16000 pounds. It is eight wheel powered and hydrostatically driven. The direction and distance travelled are provided by a Land Navigation System (LNS). The LNS measures direction as an absolute angle, for example, azimuth is measured with respect to true North. Sample readings of the LNS are provided in the appendix.

3 Description of the Data

The collection sites for images were located in the area surrounding the Martin Marietta plant in Denver. Images were acquired by moving the ALV on rough terrain (off-road), unpaved road (dirt-road) and macadamized road (on-road).

The LNS system was activated while the vehicle was in motion in order to determine the camera egomotion parameters. In addition, the initial location of the vehicle and an adequate number of environmental points were measured with the help of two theodolites. It should be noted that for *stop and shoot* the distance travelled between successive stops is not always the same. This is not a liability because we are able to determine the location of the vehicle in 3-D with the help of the LNS or survey data or both. This provides us with ground truth for the translation and rotation of the vehicle. Table 1 gives details of the images which were collected. We show respresentative images from four of the sites in Fig 2. In each case red, green and blue images of 512 x 512 resolution were acquired.

4 Analysis of the data

For the stop and shoot sequence at Site 1, a 2-D map constructed from the theodolite measurements and with the vehicle location in each frame determined by the LNS data is shown in Figure 4. The depth values of the surveyed environmental points are shown in Table 2. The corresponding points in the image are shown in Figure 5. Although we only present a 2-D map, we are in the process of making full 3-D maps of each site and of the vehicle trajectories. It should be noted that in every case we have determined the initial vehicle position by using the theodolites. Hence for every image sequence the 3-D position of the camera at each frame can be determined by using the LNS data starting from the initial vehicle position. Note that the translation and rotation parameters are also given. It is encouraging to note that the laser range finder readings for distances and the distances computed by taking theodolite readings (angles) agree to within 0.1 m for most objects.

For one of the rifle range stop and shoot sequences we computed the 3-D position of three points at every vehicle position from theodolite measurements. From this we computed the Euler angles for rotation of the LNS coordinate system. We compared this with the Euler angles for rotation computed from the LNS data. It was found that the rotation around the z-axis (Refer Fig. 3) agreed to with 0.1 degree. This is expected because the horizontal measurements of the theodolites were large and hence very reliable. The vertical measurements of the theodolites were much smaller. Nevertheless, the rotation around the x-axis and y-axis also agreed to within 0.5 degree. We have also compared the distance travelled as measured by the theodolites and laser range finder with the LNS distance travelled and found agreement to within 0.1 meter. These figures almost certainly reflect the limitations of our survey work and not the LNS itself. The manufacturer's specifications for the LNS indicate an accuracy of 0.005 degrees in pitch, yaw and roll.Our exercise in independent verification was merely designed to reassure ourselves that the LNS was functioning properly. The camera position can be obtained from the LNS position through a rigid transformation.

5 Intended use of the data

The data shall be put to the following uses:

- To recover depth of environmental objects through the application of various motion algorithm. For a detailed description of the principal algorithms of interest developed by the University of Massachusetts please refer to [1,10,11,12,13].

- Determine the accuracy of the LNS system [14] available on the ALV by comparing the results of camera egomotion parameters obtained by the LNS with theodolite measurements. This will allow us to determine whether better instrumentation is required on the vehicle. In particular our goal is to determine whether more expensive inertial navigational systems are required for ground truth or the existing LNS system is adequate.

- Comparison of results obtained by stop and shoot with move and shoot. This is of importance because in the past practically all motion data has been obtained through stop and shoot. We would like to know to what degree stop and shoot is representative of real vehicle motion.

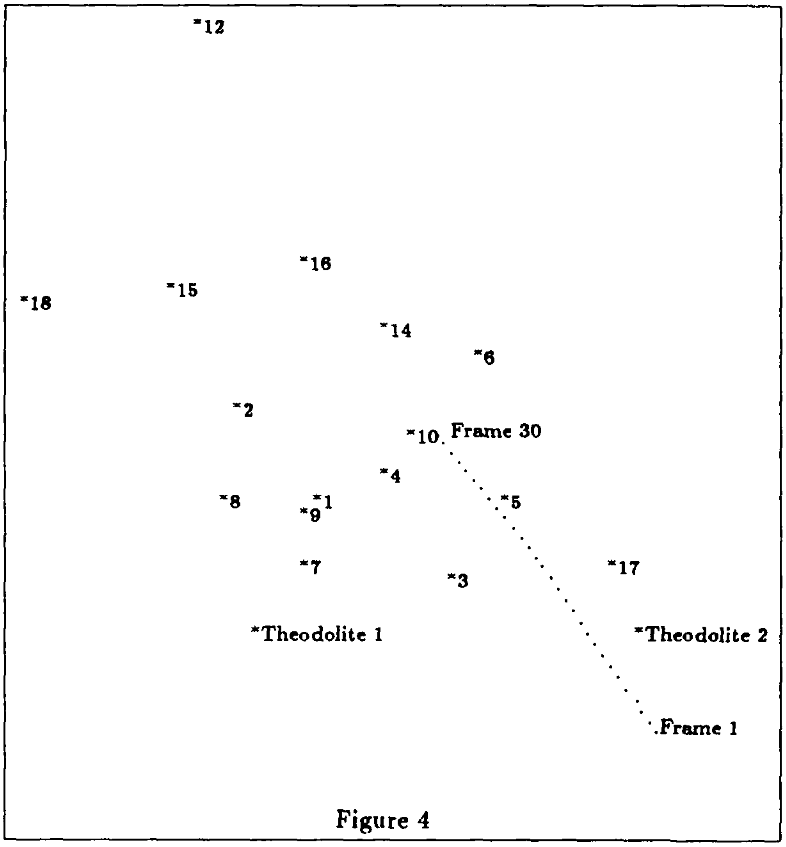
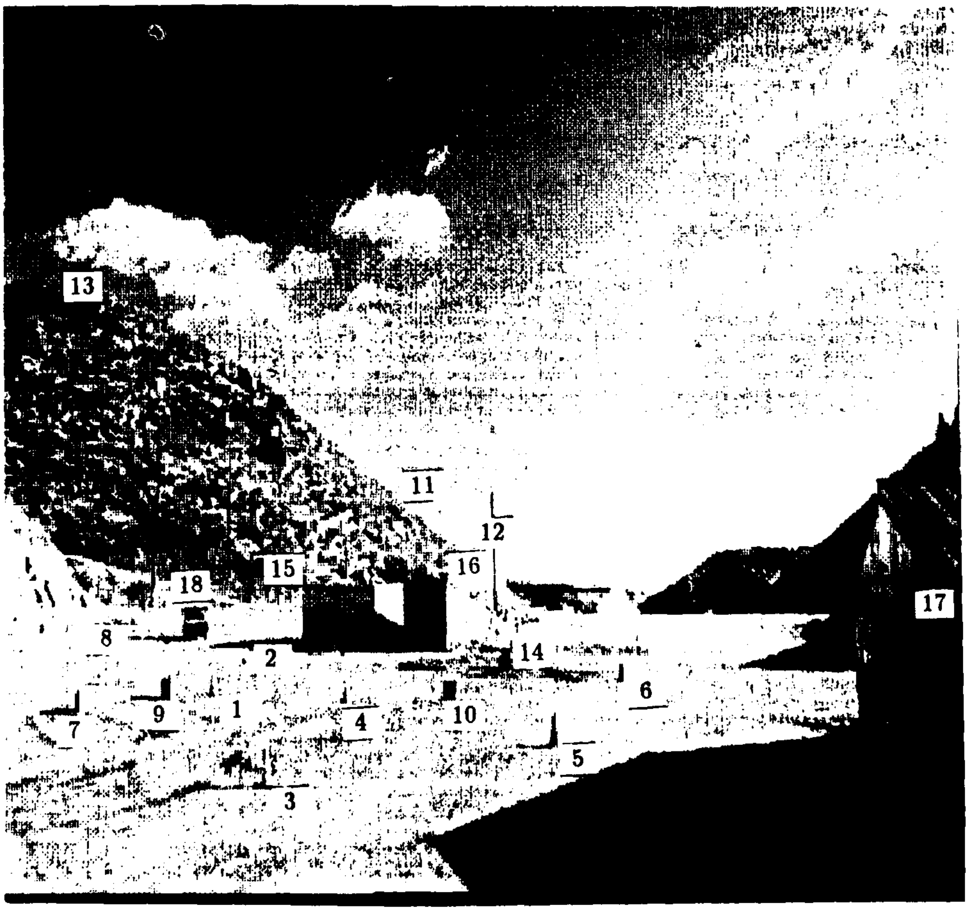


Figure 4

Pt.	Object	x	y	Dist.
1	Cone 1	-23.9	10.1	26.0
2	Cone 2	-30.3	17.3	34.9
3	Cone 3	-14.3	3.7	14.8
4	Cone 4	-19.6	12.4	23.2
5	Cone 5	-10.5	10.2	14.7
6	Cone 6	-12.5	21.2	24.6
7	Box 1	-25.1	5.6	25.8
8	Box 2	-31.2	10.4	32.9
9	Box 3	-25.0	8.7	26.5
10	Box 4	-16.9	14.7	22.4
11	Pole 2	-107.6	151.2	185.6
12	Pole 3	-32.5	46.1	56.4
13	Mountain	-171.0	86.9	191.9
14	Trash Can	-18.7	22.7	29.5
15	Bldg. 1	-35.3	25.7	43.7
16	Bldg. 2	-25.3	27.7	37.5
17	Bldg. 3	-1.7	5.1	5.4
18	Truck	-45.5	24.9	51.8

Table 2

Figure 5:



6 Conclusion

This data collection effort attempts to satisfy the longstanding need of motion researchers for outdoor motion sequences with accurate ground truth. The data is being made available to the general community and can be obtained by communicating with Ms. Valerie Cohen at UMass (E-mail address is vcohen@cs.umass.edu).

7 Acknowledgements

We wish to thank Jim Allison, Laura Haber, and Linda of Martin Marietta for their assistance in gathering data with the Autonomous Land Vehicle.

8 Appendix - The LNS data

The following is a sample of LNS data for the image sequence analysed in this paper.

```
  Acquisition time: 11800          Acquisition time: 14441
        x position: 0.000000             x position: 0.626390
        y position: 0.000000             y position: 0.502460
               yaw: 0.887852                    yaw: 0.894659
             pitch: 6.275386                  pitch: 6.275961
              roll: 0.006999                   roll: 0.007382
distance travelled 0.000000      distance travelled: 0.803084
```

References

[1] R. Dutta, R. Manmatha, E. Riseman, and M. Snyder. Issues in extracting motion parameters and depth from approximate translational motion. *Proceedings Image Understanding Workshop*, 2:945–960, April 1988.

[2] R. Y. Tsai and T. S. Huang. *Three Dimensional Motion and Structure from Image Sequences.* New York: Springer-Verlag, 1983.

[3] J. L. Barron, A. D. Jepson, and J. K. Tsotsos. The sensistivity of motion and structure computations. *Proceedings sixth national conference on Artificial Intelligence*, pages 700–705, 1987.

[4] J. Q. Fang and T.S. Huang. Solving three dimensional s ..all-rotation motion equations. *Proceedings Computer Vision and Pattern Recognition*, pages 253–258, June 1983.

[5] R. Y. Tsai and T. S. Huang. Uniqueness and estimation of 3-d motion parameters and surface structures of rigid objects. *Image Understanding 1984*, pages 135–171, 1984.

[6] M. A. Snyder. Uncertainty analysis in image measurements. *Proceedings of the DARPA Image Understanding Woikshop, Los Angeles, California*, February 1987.

[7] R. Manmatha, R. Dutta, E. Riseman, and M. Snyder. Issues in extracting motion parameters and depth from approximate translational motion. *Proceedings IEEE Workshop on Visual Motion, Irvine, Calif.*, March 1989.

[8] Gilad Adiv. Inherent ambiguities in recovering 3-d motion and structure from a noisy flow field. *Proceedings Computer Vision and Pattern Recognition*, 1985.

[9] Gilad Adiv. Inherent ambiguities in recovering 3-d motion and structure from a noisy flow field. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, to appear.

[10] Gilad Adiv. Determining three-dimensional motion and structure from optical flow generated by several moving objects. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 7(4):384–401, July 1985.

[11] D. T. Lawton. Processing translational motion sequences. *Computer Graphics and Image Processing*, 22:116–144, 1983.

[12] P. Anandan. A unified perspective on computational techniques for the measurement of visual motion. *ICCV*, pages 219–230, 1987. London, England.

[13] Lance R. Williams and Allen R. Hanson. Translating optical flow into token matches and depth from looming. *ICCV*, 1988. Tampa, Florida.

[14] University of Massachusetts and Martin Marietta. Experiment plan. *Martin Marietta*, November 1988.

MODELING SENSOR DETECTABILITY WITH VANTAGE GEOMETRIC/SENSOR MODELER[1]

Katsushi Ikeuchi and Jean-Christophe Robert
School of Computer Science
Carnegie Mellon University
Pittsburgh, PA 15213

ABSTRACT

This paper describes a method of modeling sensor detectability and its implementation in the VANTAGE geometric/sensor modeler.

Each sensor consists of one or more light sources and TV cameras, which illuminate and observe, respectively, various portions of faces of an object. Sensor detectability, which specifies where the sensor can "see" (detect), can be described by a result of AND/OR operations between illuminated and camera-observable portions of 3D faces.

In order to represent this sensor detectability, first we will define a G-source, an abstract sensor component, representing either a light source or a TV camera. Then, we will investigate G-source illumination conditions to describe under what condition each G-source illuminates (or observes) surface portions with respect to its illumination direction. We will show that the sensor detectability can be decomposed into its component G-sources' illumination conditions and AND/OR operations between them. Based on these findings, we will propose the sensor composition tree. The tree consists of G-source illumination conditions as its leaf nodes and set operations which indicate a combination of the component G-source illumination conditions, as its branch nodes. We will also propose several 2D structures to represent detected 2D appearances given by VANTAGE using a sensor composition tree. Finally, we will show how to use these capabilities in model-based vision.

INTRODUCTION

Geometric modelers allow users to create, store, and manipulate models of three-dimensional (3D) solid objects [Req80,RV82,Mor85]. These geometric modelers have found many applications in CAD/CAM and robotics areas. Such examples include mechanical part designs [Sut63,BG82], off-line robot simulations [Loz82,SM86], automatic creation of programs for numerically controlled (NC) machineries [Bez72], finite element analysis.

Building a model-based vision system based on a geometric modeler [BH86,HH87,Ike87] is one of the most interesting applications. The relevant knowledge of an object required for recognition is extracted from its representation in a geometric modeler and then used for recognition by a vision program. It has become very common to design an object by means of a geometric modeler; this designing process provides a geometric representation of an object. This geometric representation can be used to build a model-based vision system. If we can establish a systematic method of converting those representations into a recognition program of that object, recognition programs can be relatively easily obtained.

A geometric modeler represents a 3D object, while a recognition program observes a 2D appearance of the object, and thus requires its 2D representation. Moreover, model-based vision systems often use various active sensors to obtain visual information: an active sensor projects a special set of lights onto an object and measures depth/surface orientation from the returned light. Under these active sensors, the object appearances are determined by the *product* of an object model with a sensor detectability, which defines where the sensor can "see" (detect) [IK89]. As shown in Figure 1, the same object under the same attitude, thus represented as the same object model, can create different appearances (and features) when detected by different sensors. The edge-based binocular stereo reliably detects depth at edges perpendicular to the epipolar lines, which are the lines parallel to the line connecting its two TV cameras. The photometric stereo, which consists of three light sources and one TV camera, detects surface orientations of the surfaces which are illuminated by the three light sources and are visible to the TV camera. The

[1] This research was sponsored by the Defense Advanced Research Projects Agency, DOD, through ARPA Order No. 4976, and monitored by the Air Force Avionics Laboratory under contract F33615-87-C-1499.

The vi-ws and conclusions contained in this document are those of the authors and should not be interpreted as representing the official policies, either expressed or implied, of the Defense Advanced Research Projects Agency or of the U.S. Government.

light-stripe range finder, which consists of one light source and one TV camera, detects distances of surfaces which are illuminated by the light source and visible to the camera.

Thus, it is necessary for a geometric modeler to represent not only an object model but also a sensor detectability in order for it to be fully used for a model-based vision system. It is also important to represent the 2D appearance of an object explicitly and symbolically, because such a 2D representation is the one to be used by the vision system, and thus should be well-organized and easily accessible. Surprisingly, however, little effort has been expended in this direction, even though some of the early applications of the geometric modeler were vision applications [Rob65,Bau72]. This is probably because the main application of the geometric modeler is still in designing mechanical objects, and the main concern is how to represent 3D rather than 2D information.

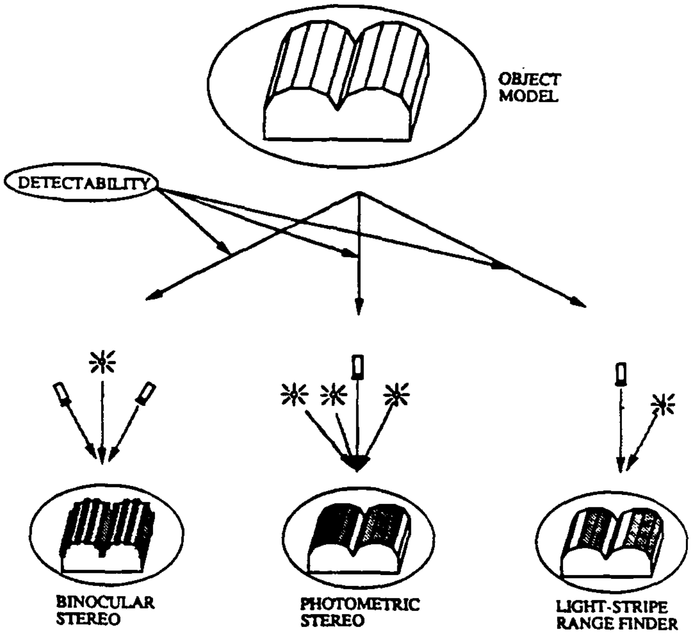


Figure 1: Object Appearances.

This pape; proposes to represent sensor detectability and to produce symbolic 2D representations of an object appearance using a geometric modeler. First, we examine how to specify the sensor detectabilities. We propose to represent them using a tool called *a sensor composition tree* and consider how to implement this sensor composition tree in our geometric modeler, VANTAGE. Then we will propose several 2D structures to represent an object appearance symbolically. We will also discuss some of the applications of this system.

DEFINITION OF SENSOR COMPOSITION TREE

Each sensor consists of one or more light sources and TV cameras, which illuminate and observe, respectively, various portions of 3D faces. Sensor detectability, which specifies where the sensor can detect, can be described as a result of AND/OR operations among illuminated and camera-observable portions of 3D faces.

In order to represent this sensor detectability, first this section defines a G-source as an abstract sensor component representing either a light source or a TV camera, and then, investigates G-source illumination conditions to describe under what condition each G-source illuminates (or observes) surface portions with respect to its illumination direction. This section also proposes a sensor composition tree, which consists of G-source illumination conditions as its leaf nodes and set operations which indicates the combination of the component G-source illumination conditions, as its branch nodes.

FEATURE CONFIGURATION SPACE

Sensor detectability depends upon various factors which include: position of a feature, orientation of a feature, reflectivity of a feature, transparency of air, ambient lighting, and so forth. In most object recognition problems the orientation and position of a feature are the most important factors. This subsection proposes a method of representing the angular freedom between a feature and a sensor, which affects sensor detectability the greatest. Later on, we will consider the effect of distance.

In order to specify the angular freedom explicitly, we attach a coordinate system to each point of an object feature and consider the relationship between the sensor coordinate system and the feature coordinate system. For example, on a 3D face, we define a coordinate system so that the z axis of the feature coordinate system agrees with the surface normal at that point and the x-y axes lie on the face, but are defined arbitrarily otherwise. For other features, we can also define a feature coordinate system appropriately. See Appendix A for more details.

Since angular relationships between the two coordinate systems are relative, for the sake of convenience we fix the sensor coordinate system and discuss how to specify feature coordinates with respect to it. The angular relation between the sensor coordinate system and feature coordinate system can be specified by three degrees of freedom: two degrees of freedom in the direction of the z-axis of the feature coordinate system, and one degree of freedom in the rotation about the z-axis. See Figure 2 (a).

Since we want to consider the angular relationship, we can translate the feature coordinate system so that the two coordinate systems share the origin. We will then define a sphere whose origin is the origin of the sensor coordinate system, and whose north pole is the z axis of that system. We will specify a feature coordinate system as a point on the sphere. Referring to Figure 2 (b), the point on the north pole of the sphere represents the feature coordinate system aligned completely with the sensor coordinate system. Any other point on the spherical surface represents a feature coordinate system obtained by rotating the sensor coordinate system around the axis perpendicular to a plane given by the sphere center, the spherical point, and the north pole until the direction from the sphere center to the point coincides with the z axis of the feature coordinate system. Figure 2 (c) represents various sensor coordinate systems corresponding to spherical points. As for a point inside of the sphere, the distance from the spherical surface to the point represents the angle of rotation (modulo 360°) around the z axis from the coordinate system corresponding to the surface point to the coordinate system corresponding to the inside point. Figure 2 (d) shows those coordinate systems corresponding to points on a radial axis. We will refer to this sphere as the feature configuration space, and represent the ability of a given sensor to detect in it.[2,3]

G-SOURCE ILLUMINATION CONDITION

Each sensor consists of two kinds of components: light sources and TV cameras. For example, both a time-of-flight range finder and a light-stripe range finder have one light source and one TV camera. The binocular stereo has one light source (without light sources you cannot observe anything) and two TV cameras; the photometric stereo has three light sources and one TV camera.

This paper regards both light sources and TV cameras as generalized sources (G-sources). Each G-source has two properties: the illumination direction and the illuminated configurations. The G-source illumination direction of a light source denotes its light source direction; the G-source illumination direction of a TV camera denotes the direction of its line of sight. G-source illuminated configurations of a light source denote the collection of the feature coordinate systems in which the feature can be illuminated, provided that its illumination direction is not occluded; G-source illuminated configurations of a TV camera denote the collection of those systems in which the feature is visible to the TV camera, provided that its illumination direction is not occluded.

Thus, in order for a feature to be illuminated by a G-source, the feature coordinate system should be in the illuminated configurations, and the G-source illumination direction should not be occluded. We will call these two conditions the *illumination condition* of the G-source.

The G-source illumination direction can be represented in the feature configuration space by a radial line from the sphere center. Let us denote this G-source illumination direction as V, which is a unit vector from the origin of the feature coordinate system to the sensor coordinate system. G-source illuminated configurations can be specified as a volume in the configuration space. For example, let us suppose a light source is located at the origin of the sensor coordinate system and the origin of a feature coordinate system is on the negative z direction of the sensor coordinate system. Then, its G-source illumination direction is represented as the line segment from the sphere center to the north pole. If this G-source illuminates faces whose surface orientation is less than 90 degrees from the illumination direction, then its G-source illuminated configurations are represented by the northern hemisphere of the feature configuration space, as shown in Figure 3. In most cases, however, the area near the equator is noisy, so we exclude that area and have a spherical cone whose axis is the same as V and whose apex angle is d.[4]

[2] This representation will not create discontinuities around the north pole as opposed to the case in which Euler angles from the sensor coordinate system to the feature coordinate system are used to specify spherical points; this representation will instead create discontinuities at the center of the sphere and at the south pole. However, this is advantageous because we mainly use the area around the north pole to discuss detectability and reliability.

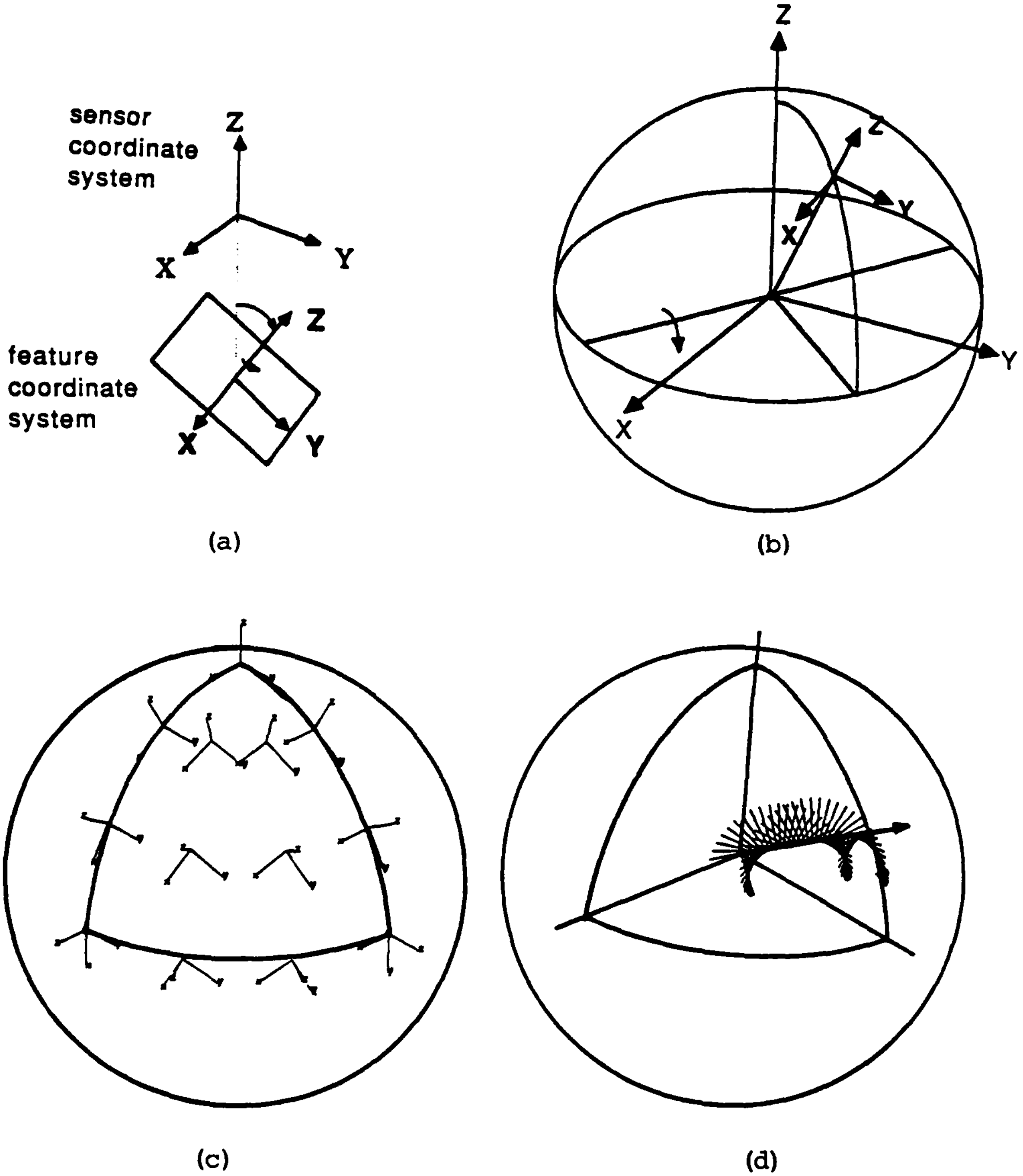


Figure 2: Feature Configuration Space.

Once a G-source illumination condition is applied to a 3D face, it generates an illuminated portion or portions and a shadowed portion or portions on the 3D face.

SENSOR COMPOSITION TREE

The detectability of a sensor, that is where a sensor can detect, can be *decomposed* into illumination conditions of its component G-sources and operations between them. For example, a photometric stereo system can detect only the portions of a 3D face on which its three light sources project light directly and which its TV camera observes. Thus, we can decompose the detectability of a photometric stereo system into four different G-source illumination conditions (three light sources and one TV camera), and *AND* operations between them. In the case of a light-stripe range finder, for a portion to be seen, it is necessary that it be illuminated by the light source and be observed from its TV camera. Thus, we can decompose the detectability into two G-source illumination conditions and *AND* operations between them.

Accordingly, portions of a 3D face detected by a sensor can be obtained by *independently* applying its component G-source illumination conditions to the 3D face and then applying set operations to the illuminated portions on the 3D face. For example, the photometric stereo's detected portions can be obtained by applying its four G-source illumination conditions to a 3D face to generate four kinds of illuminated portions on the 3D face, and then applying *AND* operations among the illuminated portions. For a light-stripe range finder, its detected portions can be obtained by applying two G-source illumination conditions to a 3D face to generate two kinds of illuminated portions on it and then applying an AND operation between the portions on the 3D face.

A TV camera has functions as G-source and as a projector. As a G-source, a TV camera generates illuminated (visible) portions on a 3D face. In other words, by tracking the lines of sight in reverse directions, we can define the illuminated (visible) portions on the 3D face. (Actually, those portions are visible from the TV camera.) Then, the TV camera projects the illuminated (visible) portions of a 3D face onto the image plane to generate its 2D appearance.

Based on this consideration, *AND, OR*, and *PROJECTION* operations are necessary for combining illuminated portions on a 3D face and for projecting, to generate a 2D appearance of the portions detected by a sensor. It is also necessary to specify the order of the set operations. For this, we will define a sensor composition tree. That is a binary tree, each of whose leaf nodes represents a G-source illumination condition and each of whose branch node represents one of the set operations: *AND,OR*, or *PROJECTION.*

We can represent several sensors' detectability using this sensor composition tree. For example, Figure 4(a) shows a light-stripe range finder and its sensor composition tree. The right leaf node, *GS1*, corresponds to the illumination condition of the light source, and the center node, *GS2*, represents the illumination condition of the TV camera. These two nodes are connected by the operation, *AND*. Since the appearance is generated with respect to the coordinate of the TV camera, the operation, *PROJECTION*, is applied with respect to *GS2.*

Two other examples, a two-light light-stripe range finder and a photometric stereo, are also shown in Figure 4.

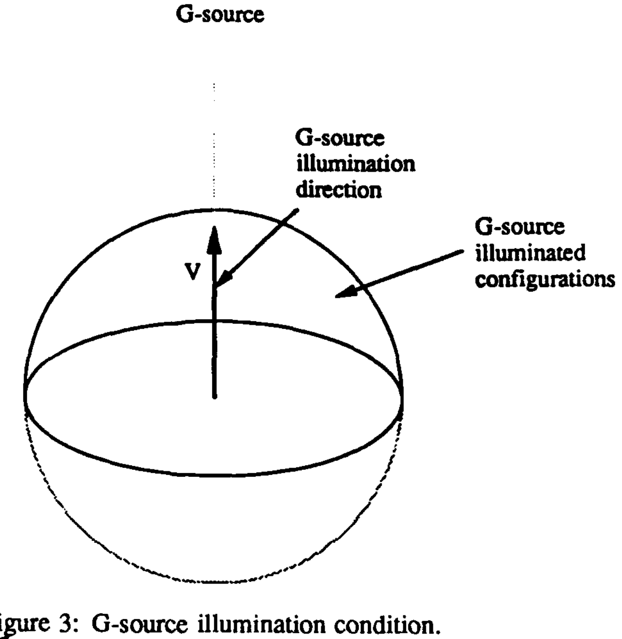

G-source
G-source illumination direction
G-source illuminated configurations
V

Figure 3: G-source illumination condition.

[3] *Most sensors detect faces. In this case, we only need to consider the spherical surface as the configuration space instead of the total sphere.*

[4] The current implementation of VANTAGE can handle only the normal G-source, whose illuminated configurations are depicted as a spherical cone, describe above. For other types of G-sources, see [IK89].

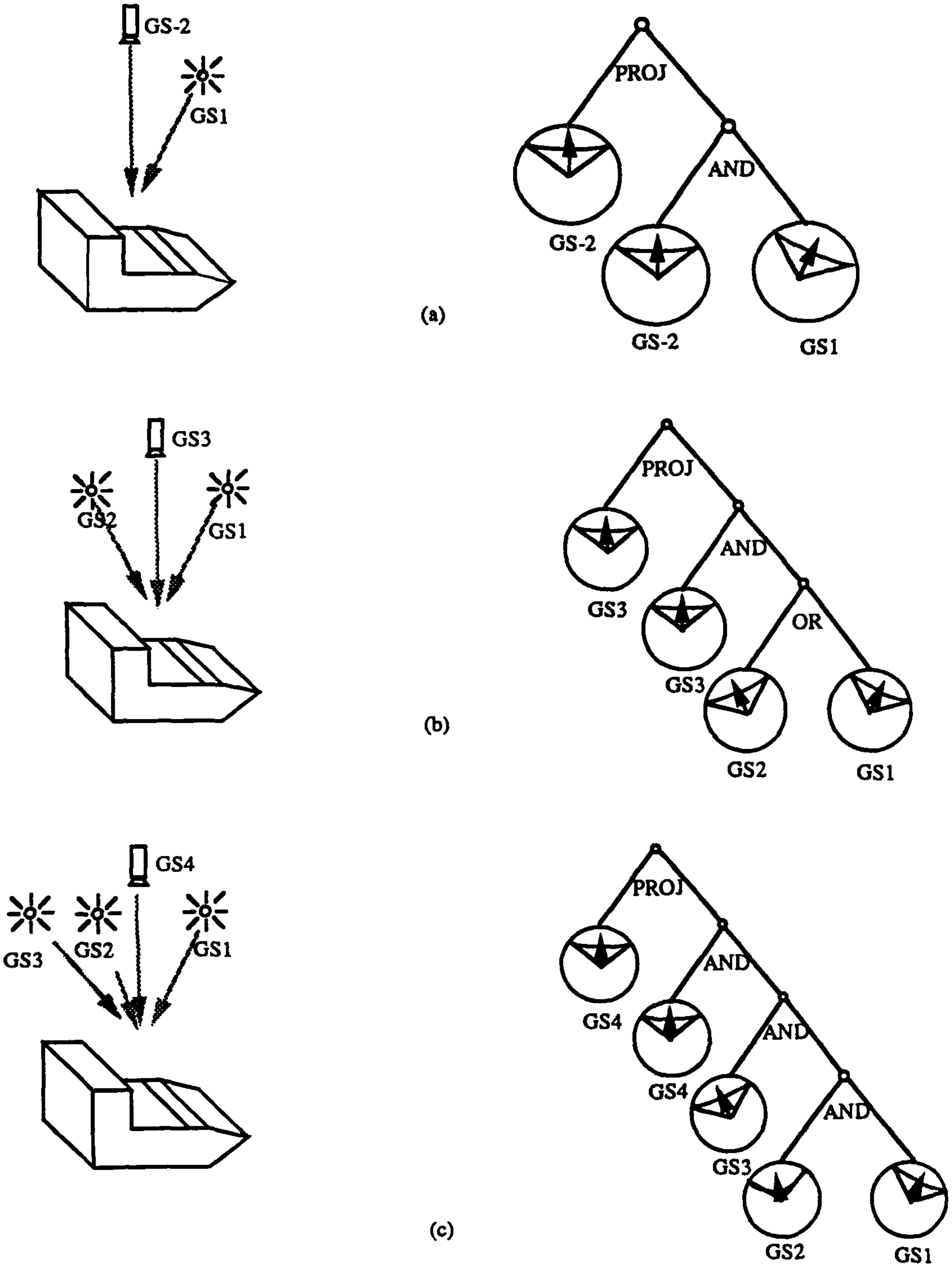


Figure 4: Example of sensor composition tree: (a) The sensor composition tree of a light stripe range finder; (b) The sensor composition tree of a two-light stripe range finder; (c) The sensor composition tree of a photometric stereo sensor.

SENSOR COMPOSITION TREE IN VANTAGE

This section considers a method of representing the sensor composition tree and the operations required to determine where a sensor can detect in VANTAGE.

G-SOURCE AND ITS OPERATIONS

This subsection defines a method of representing a G-source and its illumination condition in VANTAGE. VANTAGE represents a G-source illumination condition as a node of the sensor composition tree. A VANTAGE function, `SCTNODE`, creates a node of a sensor composition tree in VANTAGE.

The G-source illumination direction, and thus also the illuminated configurations depend on the position of the feature coordinate system with respect to the the sensor coordinate system. However, if we can assume that a G-source is far away from the object, then the G-source illumination direction becomes a constant vectors at any point of an object considered. In this case, that direction can be specified as a constant vector, $V = (X_v, Y_v, Z_v)^t$ with respect to the sensor coordinate system at any point of the object.

The G-source illuminated configurations are also represented as an invariant spherical cone regardless of the position of the feature. The axis of the cone is the same as the illumination direction, V, while the apex angle is denoted as *d*. Using these parameters, the illumination condition of the distant G-source, called the *Orthographic Generalized Source*, can be represented as

```
(SCTNODE OGS g-source-name V d)
```

in VANTAGE, where `g-source-name` is a node name used in the sensor composition tree.[5]

In general, the G-source illumination direction depends on the position of a feature. As a result, the G-source illuminated configurations change over the object. However, the shape of the spherical cone, whose apex angle is *d*, is invariant with respect to the illumination direction. Thus, in this case, called the *Perspective Generalized Source*, we will specify the explicit position of the G-source.[6]

```
 (SCTNODE PGS g-source-name X d),
```

where *X* denotes a matrix that indicates the coordinate system of the G-source with respect to the sensor coordinate system.

VANTAGE calculates the illumination direction based on the position of the feature with respect to the sensor and determines the illuminated portions of a face. These illuminated portions of the face is attached to the *3D boundary representation* of the 3D face as 3D face illumination properties.

Figure 5(a) shows an example of a 3D scene, where a 3D scene refers to a collection of several 3D objects. In this example, it consists of one cube, one arch, and one table. To this 3D scene, we will apply one sensor which consists of two G-sources: one light source from the right direction and one TV camera from the left direction. We will use this 3D scene as an illustrative example throughout this paper.

One G-source, corresponding to the light source, generates 3D illumination properties on its 3D faces, as shown in Figure 5(b), while another G-source, corresponding to the TV camera, generates other 3D illumination properties on them, as shown in Figure 5(c). Since a property does not necessarily take a uniform value over a single 3D face (for example, only one portion of a face may be shadowed while other portions may be illuminated by G-source 1), VANTAGE divides the 3D face according to the different G-sources, and creates corresponding 3D subfaces.

IMPLEMENTATION OF SENSOR COMPOSITION TREE

It is necessary to consider set operations on 3D illumination properties on a 3D face. We will define an *AND* operation between an illuminated property *I* and a shadowed property *S* by two different G-source, *i, j* as follows:

$$\begin{aligned} I &\leftarrow (AND\ I_i\ I_j) \\ S &\leftarrow (AND\ I_i\ S_j) \\ S &\leftarrow (AND\ S_i\ S_j) \end{aligned}$$

For example, referring to Figure 6, each G-source, Light-1, 2, and 3, generates its own illuminated portion and shadow portion. Applying *AND* operations to an illuminated portion gives the result shown in Figure 6 (a). The illuminated portions can be obtained by applying *intersection operations*, as shown in Figure 6 (b). The shadowed portions can be obtained by applying *union operations*, as shown in Figure 6 (c). VANTAGE implements this *AND* operation as intersection operations through a recursive polygon clipping process. See [IR89] for more details.

[5] The current implementation of VANTAGE has also optional arguments to indicate *type, focal length*, and *image size*.

[6] More precise specification is possible such as defining focal length. See [BRH*89] for more details.

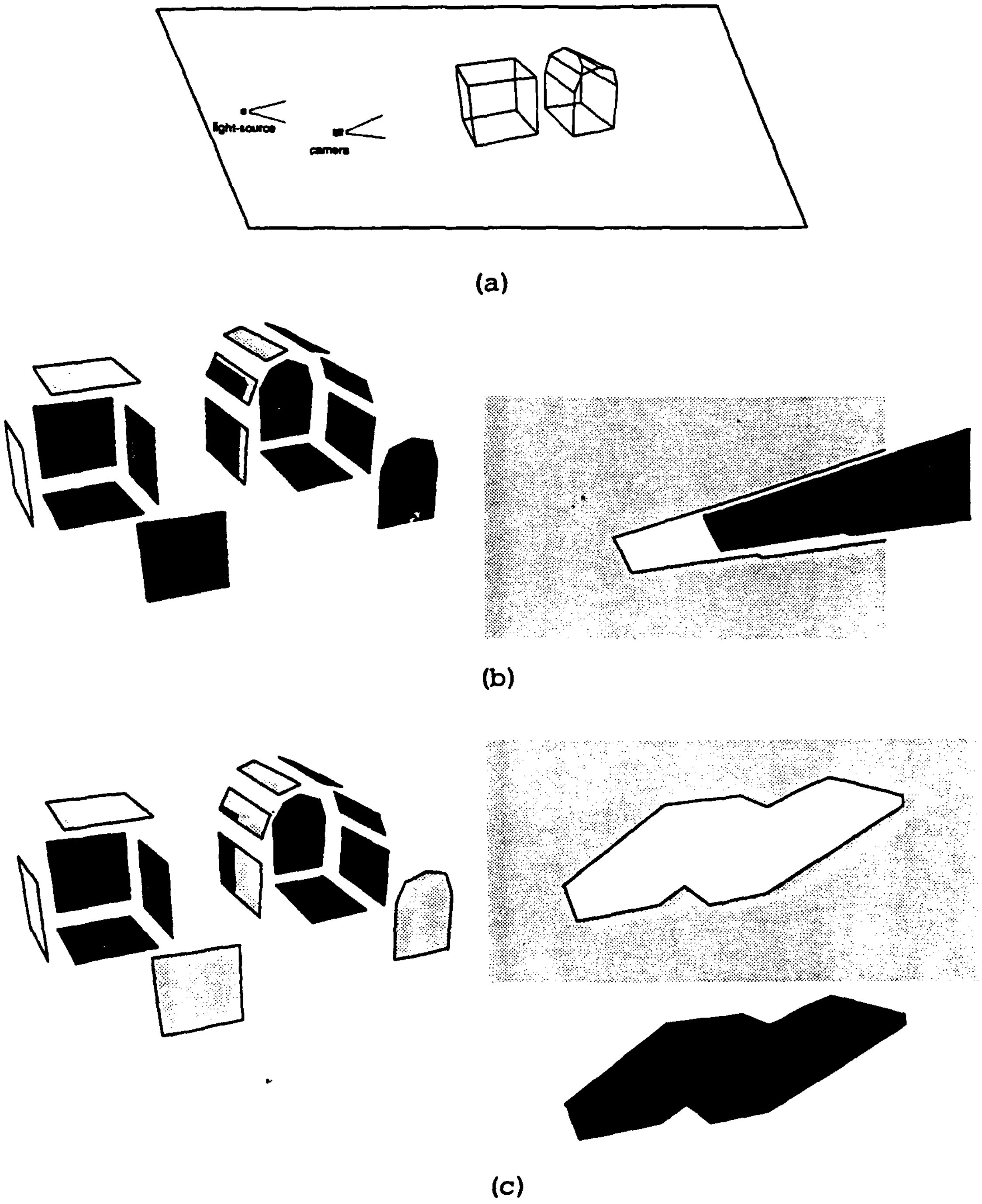


Figure 5: 3D face properties: (a) 3D scene consists of one cube, one arch, and one table. A sensor, which consists of two G-sources, one light source from the right direction and one TV camera from the left direction, is applied to the 3D scene; (b) 3D illumination properties generated by the light source; (c) 3D illumination properties generated by the TV camera.

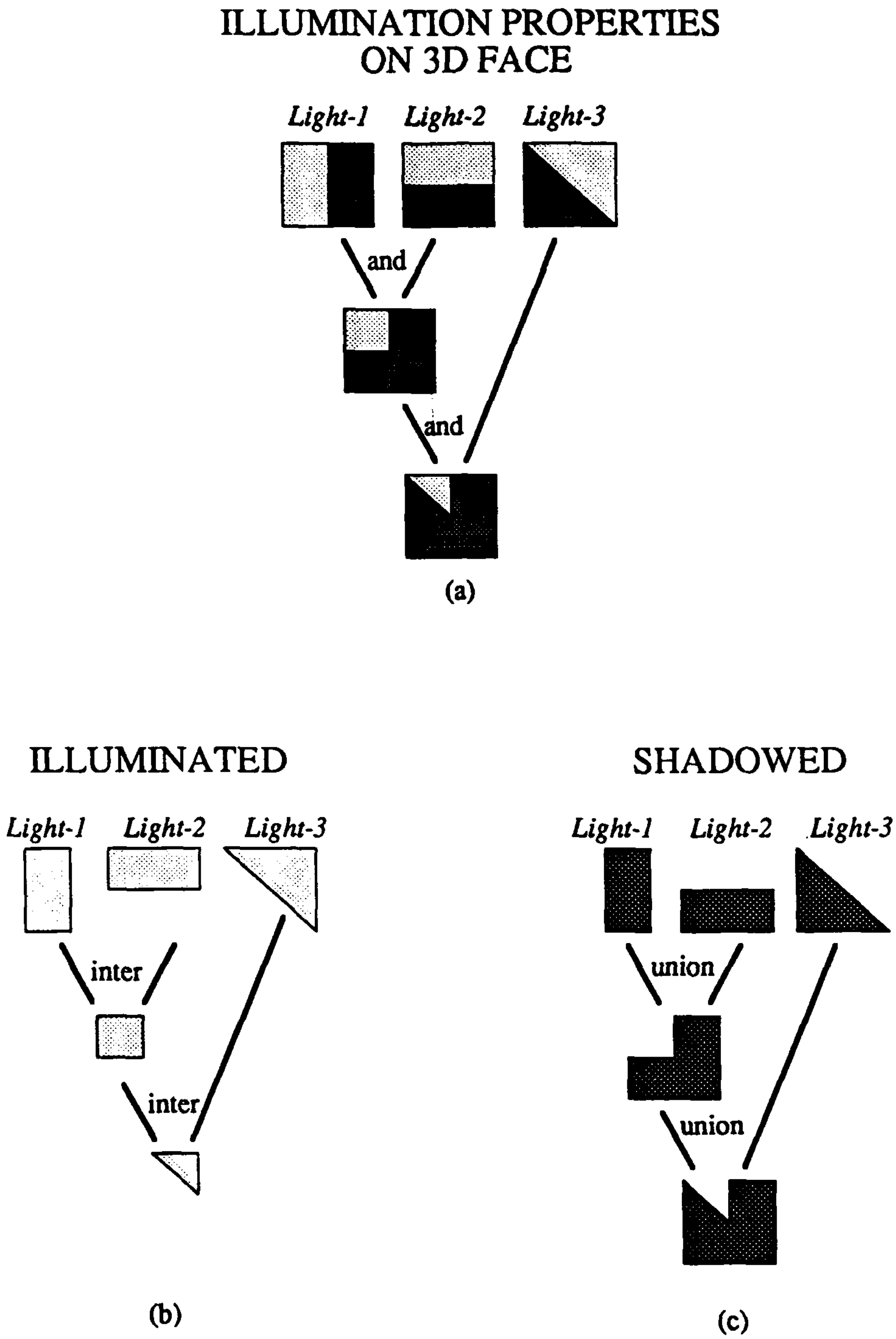


Figure 6: AND operation among illuminated portions.

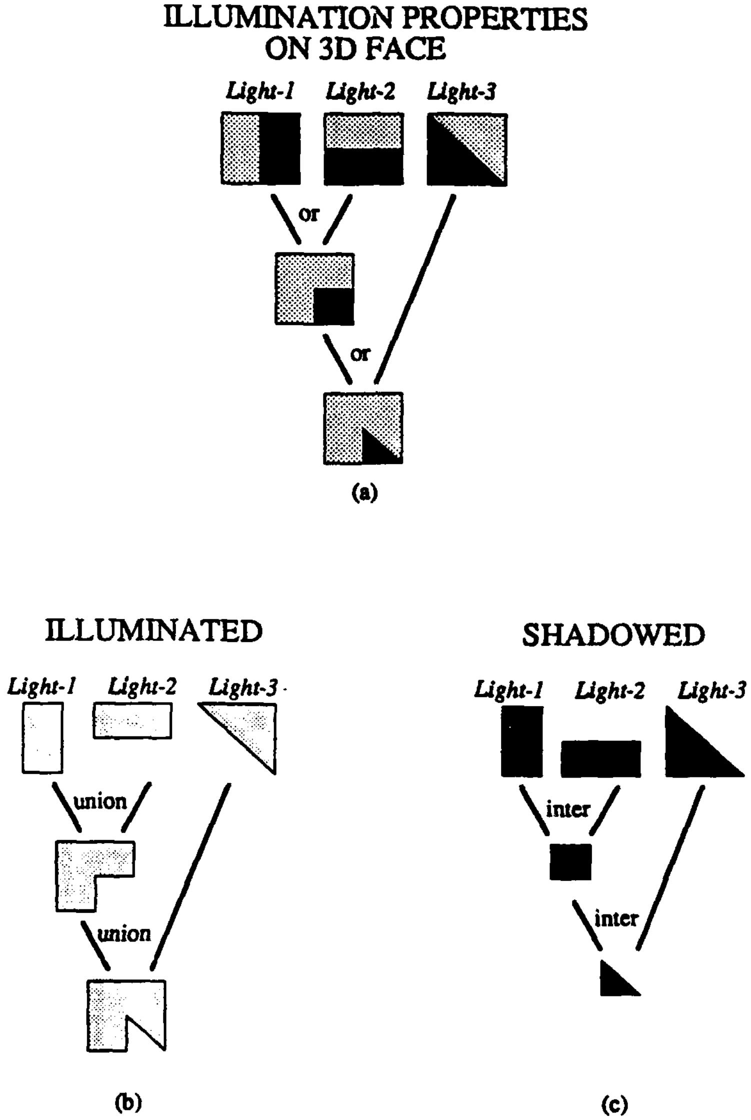
We define *OR* operations as follows:

$$I \leftarrow (OR\ I_i\ I_j)$$

$$I \leftarrow (OR\ I_i\ S_j)$$

$$S \leftarrow (OR\ S_i\ S_j)$$

The operation *OR* has a complementary relationship with the *AND* operation. Let us suppose the same three light sources generate three illuminated portions on the same 3D face and we need a part of *OR* of the three illuminated areas. See Figure 7 (a). The illuminated portion can be obtained by applying *union* operations to the illuminated portions, as shown in Figure 7 (b), while the shadowed portion can be obtained by applying *intersection* operations to the shadow portions.



(a)

(b)

(c)

Figure 7: OR operation among illuminated portions.

Let us consider the case where we generate an appearance of a 3D face, from a TV camera, of a portion illuminated by a different light source. We will generate only a portion that is illuminated and visible. The operation *AND* is executed between the illuminated portion and visible portion on the 3D face. Then, that portion is projected using the operation, *PROJECTION*, onto the image plane of the TV camera.

These set operations are defined as nodes of the sensor composition tree. In order to create such nodes, we use

```
(SCTNODE operation node-name left-node right-node)
```

where operation is one of the following:

- AND – AND operation is applied between two illuminated portions.
- OR – OR operation is applied between two illuminated portions.
- PROJECTION – a projection is generated from a 3D face.

Once a sensor composition tree is represented, it is applied to a 3D scene by a function called `apply-sensor-to-scene`. This function is executed in three steps. In the first step, portions illuminated by each G-source on a 3D face are generated and stored as 3D illumination properties on the 3D face. These properties define how to divide each 3D face into several subfaces with the same properties. In the second step, set operations, such as the *AND* and *OR* operation are executed between illumination properties along the sensor composition tree, and the results are also stored on a 3D face. In the third step, visible illuminated and visible shadowed portions of 3D faces are projected onto the image plane as its 2D appearance.

The resulting 2D appearance consists of visible 2D faces, which are divided into illuminated portions and shadowed portions. Strictly speaking, the sensor composition tree should give only illuminated portions, and thus, the resulting 2D appearance should consist only of illuminated portions. However, depending on applications, we may be interested in shadowed portions. Thus, the current implementation of VANTAGE will produce shadowed portions as well as illuminated ones.

TWO-DIMENSIONAL STRUCTURE

This section examines the representational structure of 2D appearances. Before discussing the main topic, we will review 3D representation of VANTAGE [KBR*88]. Then, we will examine several 2D structures having various priorities regarding illumination information with respect to the 3D structure.

3D STRUCTURE

We will begin by considering the definition of a 3D face. Several definitions of faces are possible depending on continuity conditions across their boundaries. Referring to Figure 8, if we define a face as the surface patch such that across its boundary C^0 continuity is violated, then, the entire boundary of an object becomes one face. VANTAGE refers to this level as .he *Solid level.*

We can define another type of face as the surface portion such that across its boundary, at most, C^1 continuity is violated. This level, called as *Merged level*, generates faces as shown in Figure 8(b). Note that the cylindrical part and the planar part are connected. We can also define C^2 faces as shown in Figure8(c). This level is referred to as the *Grouped level* in VANTAGE.

While Class $C^0, C^1, andC^2$ faces are defined purely mathematically, VANTAGE has one more level; at this bottom level, the boundary representation of a object contains only planar faces, which have originated from either polyhedral primitives or from polyhedral approximations of curved primitives.[7] The latter faces are referred to as approximation faces. This level will be referred to as the *APP level.*

VANTAGE builds the higher levels of representation from the APP level representation. First of all, VANTAGE can group a set of adjacent planar faces that approximate the same curved surface into one *curved* face. This grouping operation can generate a grouped level (Class C^2) representation of an object. Second, faces that are tangent across an edge, that is, C^1 continuous, are also merged into a *merged* face and the merged level (Class C^1) representation is completed. Finally, VANTAGE can group all faces that are connected and generate the solid level (Class C^0) representation. We will refer to this whole structure as the *3D structure* of an object.

VANTAGE maintains a 3D boundary representation of an object at each level. This representation contains lists of faces, edges and vertices at each level. Vertices contain their respective coordinate values, and edges join these vertices. The edges are either straight lines or curved lines, depending on the level. The faces are either planar polyhedra or curved polyhedra and are represented by a collection of connected edges at that level. The neighborhood information or *topology* relates the edges, faces, and vertices of the object, and is stored in the form of

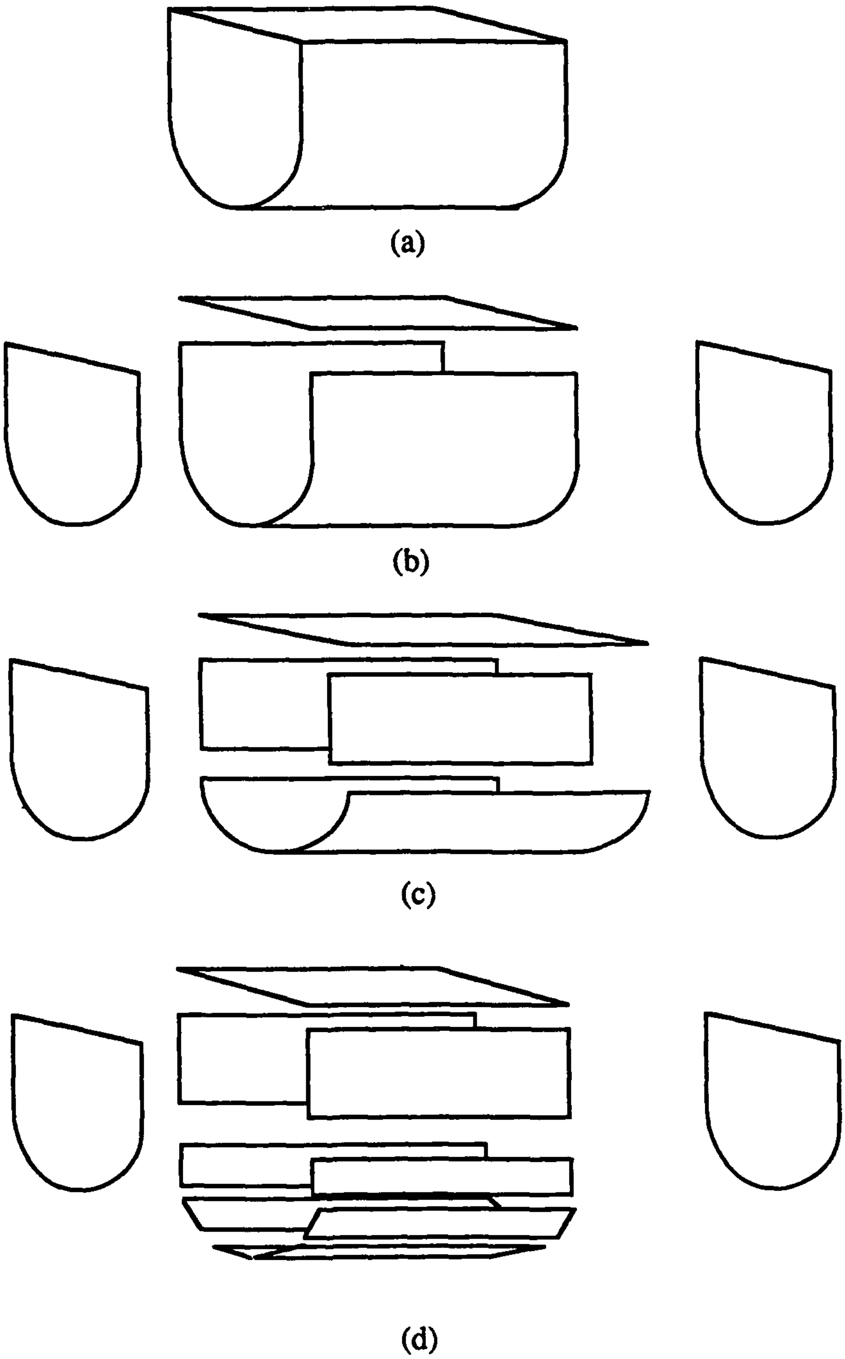


Figure 8: Definition of Faces: (a) Class C^0 face: *Solid level*; (b) Class C^1 faces: *Merged level*; (c) Class C^2 faces: *Grouped level*; (d) polygon approximation faces: *APP level face*.

winged-edge representation [Bau72], where each level maintains its own winged-edge representation. For example, the arch in the example scene shown in Figure 5(a) has the 3D structure as shown in Figure 9.

Several 2D structures are possible, depending on the priority of illumination information with respect to the 3D levels. For example, we can put the illumination information below the 2D APP level; we can also put the illumination information above the 2D solid level. In order to examine these cases, we will consider the 3D scene shown in Figure 5(a). We will assume a sensor which consists of two G-sources: a light source which illuminates the scene from the right direction and a TV camera which observes it from the left direction. Applying this sensor to the 3D scene, VANTAGE will generate a 2D appearance of it. In the following subsections, we will investigate several 2D structures of the 2D appearance.

FLAT STRUCTURE

The 2D appearance of a 3D scene is an explicit symbolic representation obtained by projecting the scene onto the image plane using the sensor composition tree. VANTAGE first generates a 2D boundary representation by projecting the 3D APP level boundary representation of the scene. Thus, a 2D boundary representation of a scene is a set of 2D faces, 2D edges, and 2D vertices, linked by winged-edge relations. Each 2D element corresponds to its 3D counterpart.

Then, VANTAGE projects the 3D illumination properties, generated along the sensor composition tree, of the 3D faces onto the 2D boundary representation. Thus, each 2D face is divided into several subfaces to correspond to illumination properties of its 3D face. Subfaces do not maintain winged-edge relations; they represent internal structure of a 2D face.

Thus, the basic 2D boundary representation is a collection of 2D APP faces, which maintain the illumination properties below them as subfaces. This representation, which we will refer to as the *flat structure*, is the default 2D representation of VANTAGE.

The current implementation of VANTAGE supports three kinds of 3D illumination properties: illuminated (I), cast-sha'owed (CS), and self-shadowed (SS). These three values can be consolidated into two values, illuminated (I) and shadowed (S) by the operation:

$$I \leftarrow I$$
$$S \leftarrow (OR\ CS\ SS).$$

This is necessary to be able to apply *AND/OR* operations to illumination properties.[8]

Figure 10(a) shows the 2D appearance of the example scene. This 2D appearance is represented by the flat structure as shown in Figure 10(b). Note that the several approximate faces are divided into two parts to store the illumination information.

ILLUMINATION-ORIENTED STRUCTURE

This structure gives priority to the illumination information. In it, a 2D appearance will be classified into two categories: illuminated, and shadowed. Within each category, its own 2D structure is constructed.[9]

First, VANTAGE classifies the 2D subfaces into the two categories. In each category, the following operations are repeated. From the 2D APP level, as is the case in 3D, VANTAGE builds three more levels of representation based on surface properties. VANTAGE can group a set of adjacent 2D subfaces which come from 3D APP faces of the same curved surface into a *2D curved* face. Second, 2D faces which come from 3D faces that are tangent across an edge, are also merged into a *2D merged* face. And finally, VANTAGE groups 2D subfaces which come from the same object into a complete 2D structure.

Figure 11 shows the illumination-oriented structure of the example scene. Note that the illuminated portion of the cylindrical part of the arch is represented in several levels.

This illumination-oriented structure is useful when applying the sensor composition tree to a 3D scene, because the sensor usually detects only illuminated portions which are in the illuminated category.

This structure is also useful when generating a 2D structure of a 3D scene of a simple camera projection. In this case, we will put the light source at the same place as the viewer. Then, all visible portions become illuminated, and thus, we can get a 2D structure of all visible portions of a 3D scene.

[7] VANTAGE represents a complicated object as a collection of simple primitive objects, such as a cube or a cylindrical object, by using CSG representation.

[8] VANTAGE has two methods of generating a 2D appearance: applying a sensor composition tree, and applying a TV camera and a light source directly. We explain the former method in this paper. If we use the latter method, we can generate a 2D appearance which also maintains three categories of illumination information: illuminated, self-shadowed, and cast-shadowed.

[9] If the 2D flat structure maintains three categories of illumination properties, then three structures are generated within the three categories: illuminated, cast-shadowed, and self-shadowed.

As mentioned before, the current implementation of VANTAGE supports three kinds of illumination properties: illuminated (*I*), cast-shadowed (*CS*), and self-shadowed (*SS*). In the sensor composition tree module, the following operations are performed on the illumination properties:

$$I_i \leftarrow I_i$$

$$S_i \leftarrow (OR\ CS_i\ SS_i),$$

where *i* denotes that the illumination property is given by the G-source *i*. However, we can also define

$$I_i \leftarrow (OR\ I_i\ CS_i)$$

$$S_i \leftarrow SS_i$$

Then, by applying *OR* operations to a 3D scene, the shadow category gives a collection of the portions represented by $(AND\ SS_i\ SS_j)$. Using this schema, we can generate 2D structures of various kinds of combinations between G-sources. This is particularly useful when each G-source has a different color and we are interested in analyzing the color of a 2D appearance.

GEOMETRY-ORIENTED STRUCTURE

We can also sc. the priority of the illumination information somewhere between the top level (solid level) of the 2D structure and the bottom level (APP level).

Let us suppose that we put the illumination information between the solid level and the merged level. Then, at each solid, a face structure is generated. In the example scene case, since there are three objects, three structures are generated: the front cube, the middle cylinder, and the back cube. Then, each structure will be divided into two parts: illuminated and shadowed. See Figure 12(a). This structure is particularly useful when we analyze illumination conditions of each object.

We can also put the illumination information between the merged level and the grouped level. Then, a merged face will be divided into three categories of illumination conditions, as shown in Figure 12(b). This structure is useful when we produce detectable faces by means of an active sensors, because the direct output of an active sensor are the collection of C^1 faces.

The illumination information can also be stored between the grouped level and the APP level. Then we will obtain the structure as shown in Figure 12(c).

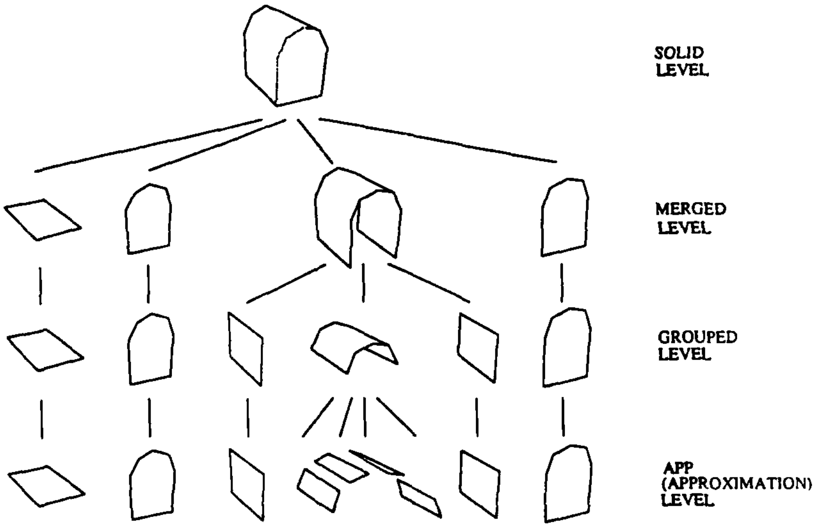


Figure 9: 3D structure.

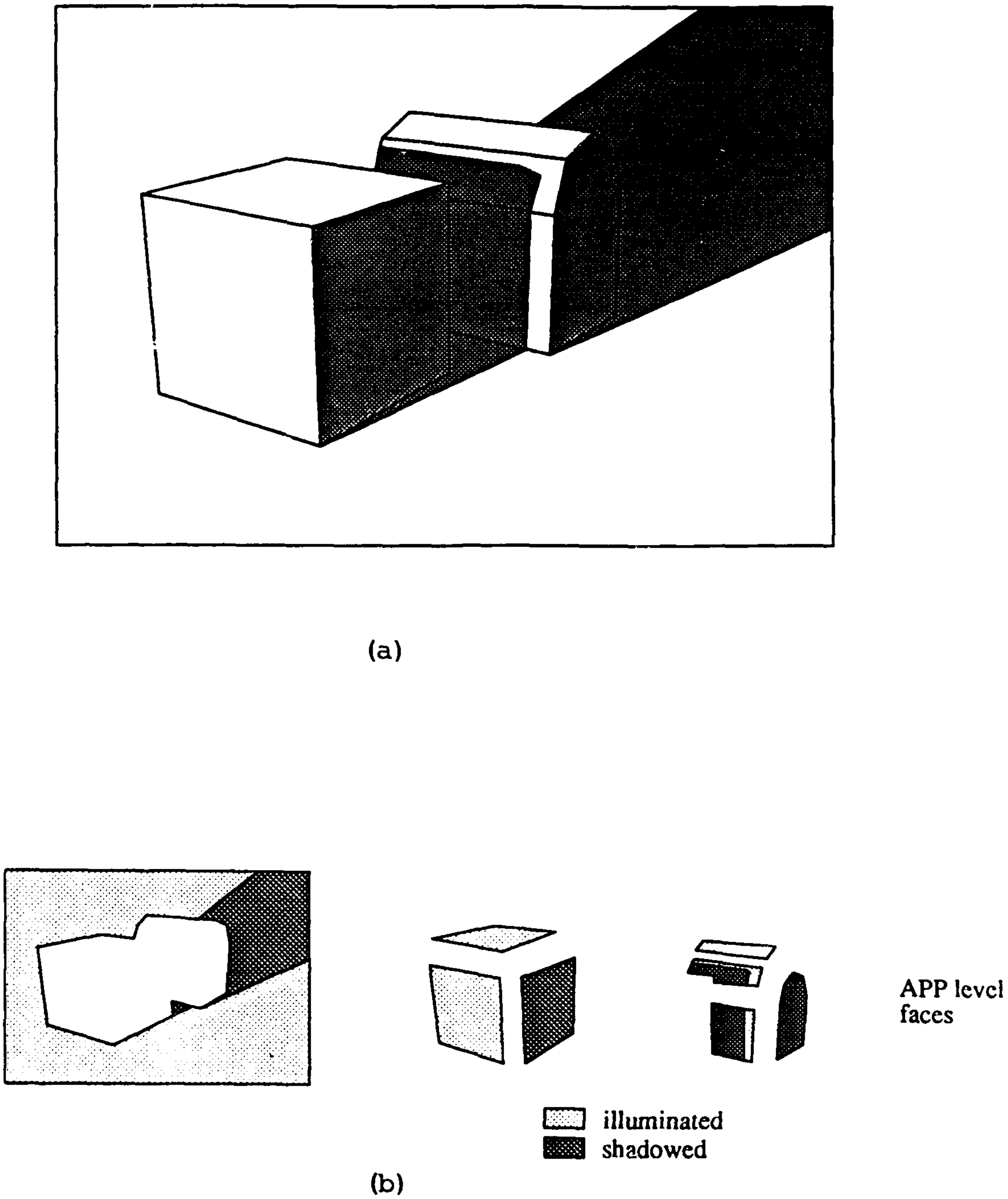


Figure 10: Flat structure: (a) 2D appearance of the example scene; (b) Flat structure of the 2D appearance.

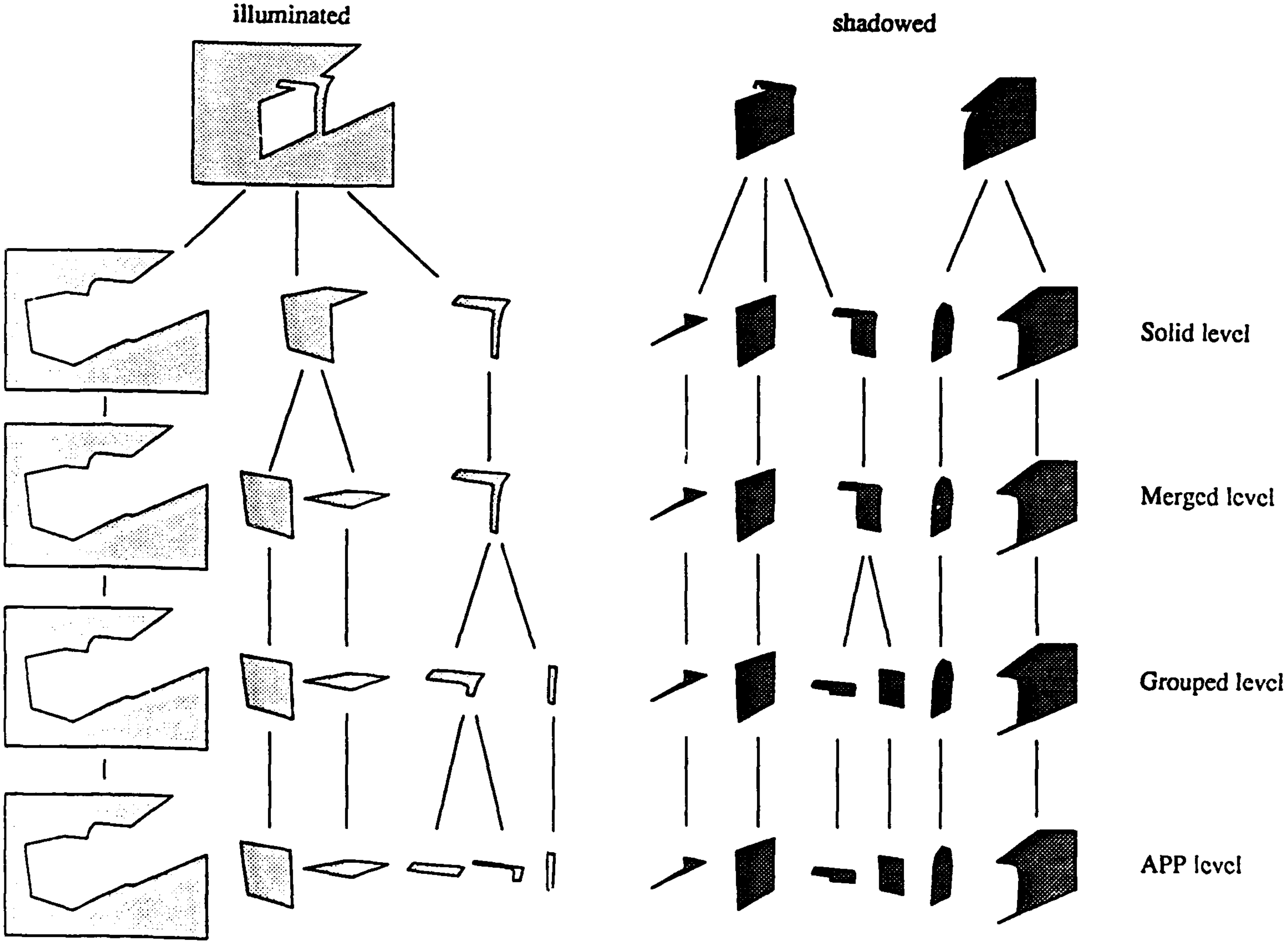
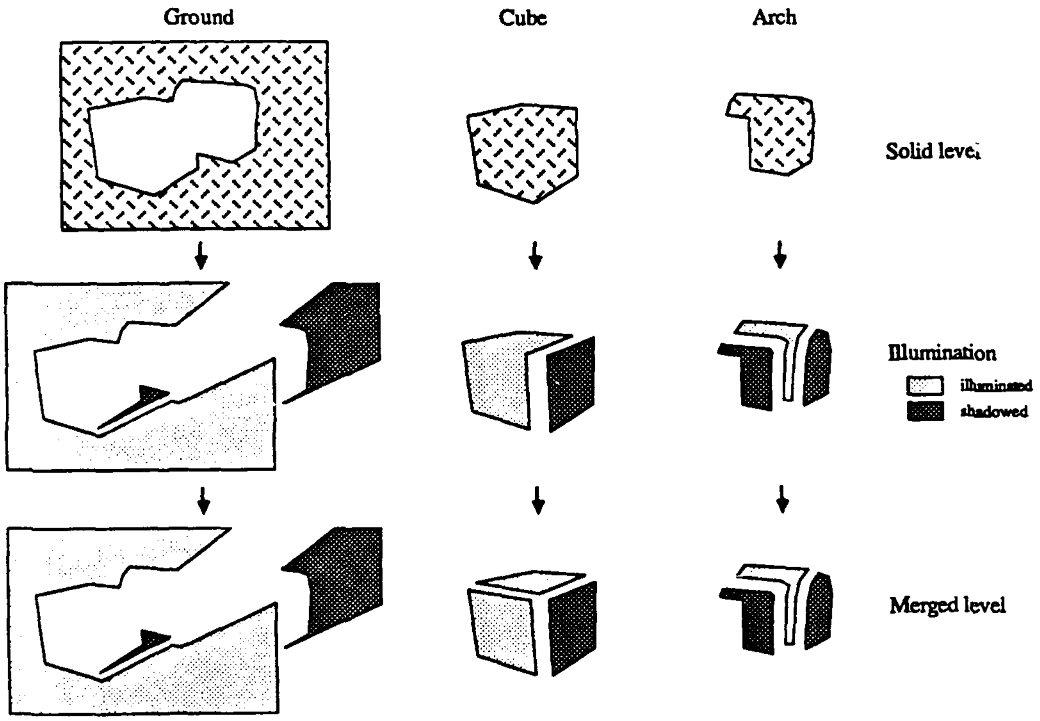
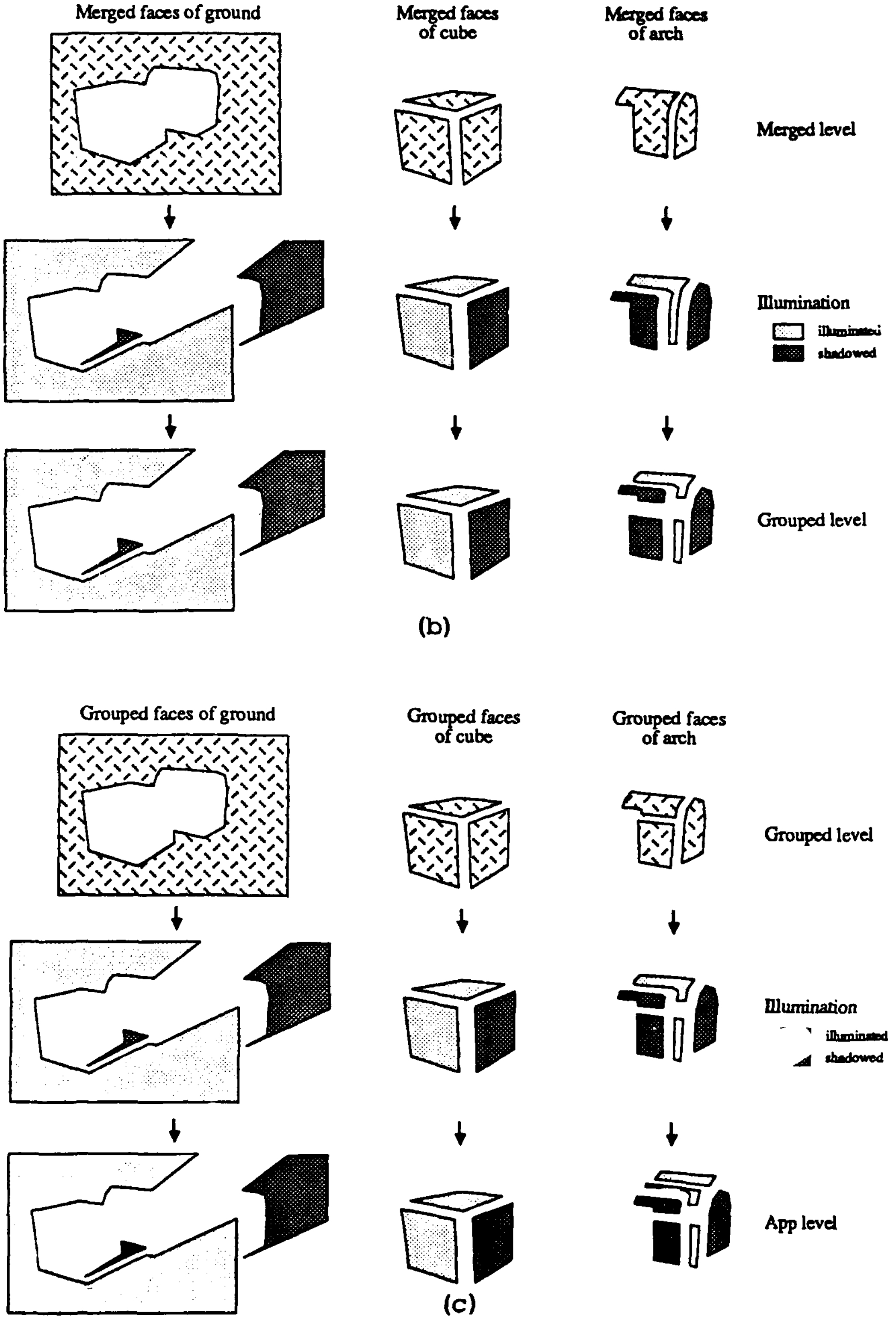


Figure 11: Illumination-Oriented Structure.



(a)

Figure 12 (cont)



(b)

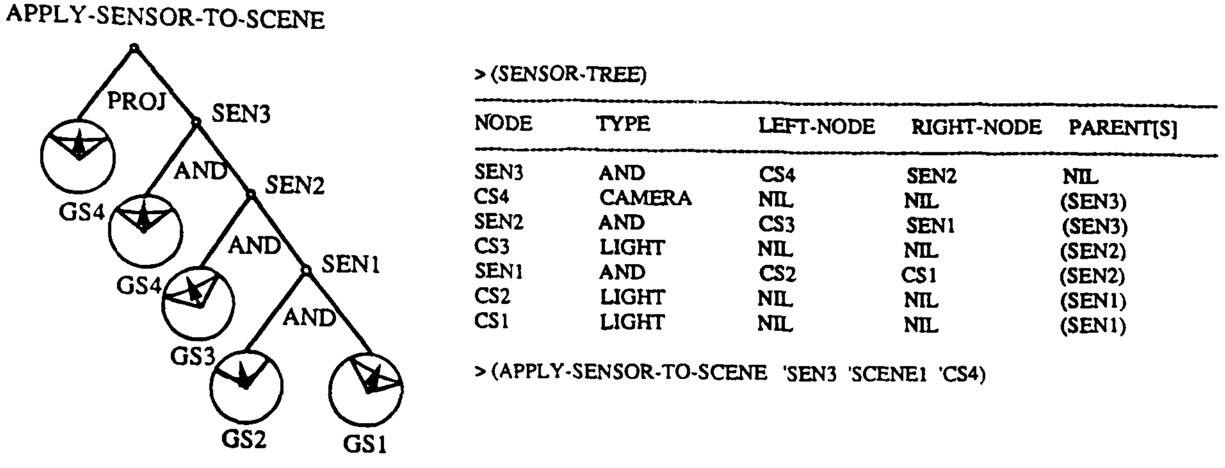
(c)

Figure 12: Geometry-oriented structure: (a) Solid-merged; (b) Merged-grouped; (c) Grouped-APP.

APPLYING VANTAGE TO MODEL BASED VISION

This section briefly outlines how to apply the VANTAGE geometric/sensor modeler to one of the applications of Model-based vision: geometric compiler project to automatically generate an object localization program from an object model.

From the 3D representation of an object in VANTAGE and the sensor detectability of a particular sensor, we can generate a possible appearance of that object under that sensor. For example, Figure 13(a) depicts a sensor composition tree of photometric stereo, while Figure 13(b) represents the internal representation of the tree in VANTACE.



```
> (SENSOR-TREE)
--------------------------------------------------------------------------
NODE     TYPE        LEFT-NODE    RIGHT-NODE    PARENT[S]
--------------------------------------------------------------------------
SEN3     AND         CS4          SEN2          NIL
CS4      CAMERA      NIL          NIL           (SEN3)
SEN2     AND         CS3          SEN1          (SEN3)
CS3      LIGHT       NIL          NIL           (SEN2)
SEN1     AND         CS2          CS1           (SEN2)
CS2      LIGHT       NIL          NIL           (SEN1)
CS1      LIGHT       NIL          NIL           (SEN1)

> (APPLY-SENSOR-TO-SCENE 'SEN3 'SCENE1 'CS4)
```

(a)

(b)

Figure 13: Sensor composition tree in VANTAGE: (a) The sensor composition tree of the photometric stereo; (b) The internal representation of the sensor composition tree. In the current implementation, the internal representation keeps only *AND/OR* operations. The final projection is combined into a `apply-sensor-to-scene` function.

Figure 14 shows an example of the application of the sensor composition tree of the photometric stereo to a car model. Figure 14 (a) is the car model. This sensor consists of four G-sources: three light sources and one TV camera. Thus, the first step in applying the sensor composition tree to the car model generates 3D illumination properties of four G-sources, from *GS1* through *GS4*. These 3D illumination properties are stored on 3D faces. Then, set operations between 3D illumination properties are executed along the sensor composition tree as shown in Figure 14(b). Finally, the projection of the car is generated with respect to the G-source *GS4*. Figure 14(d) shows the corresponding needle map of the car obtained by the photometric stereo system, while Figure 14 (c) shows the original scene. They correspond with each other quite well. As you can see in this example, we cannot predict object appearances correctly without the sensor modeling capability of a modeler like VANTAGE.

Using the illumination-oriented structure, we can obtain the portions of the car detectable by the photometric stereo. Since the appearance is representable by frames, we can easily convert the output from VANTAGE to the appearance represented in Figure 15(a). We can classify and categorize various appearances into possible aspects, where each aspect shares the same combination of detected 2D faces. Since whether a 2D face is detected depends on sensor detectability, the possible aspects also depend on sensor detectability. In other words, without this capability of VANTAGE, we cannot generate possible aspects of an object under one particular sensor. T- is, this capability is one of the most fundamental components of an automatic generation of object localization program.

One aspect structure, a symbolic representation of an aspect, is constructed at each aspect group, where an aspect is a topologically equivalent class of object appearance and is used as a basic representation for the geometric

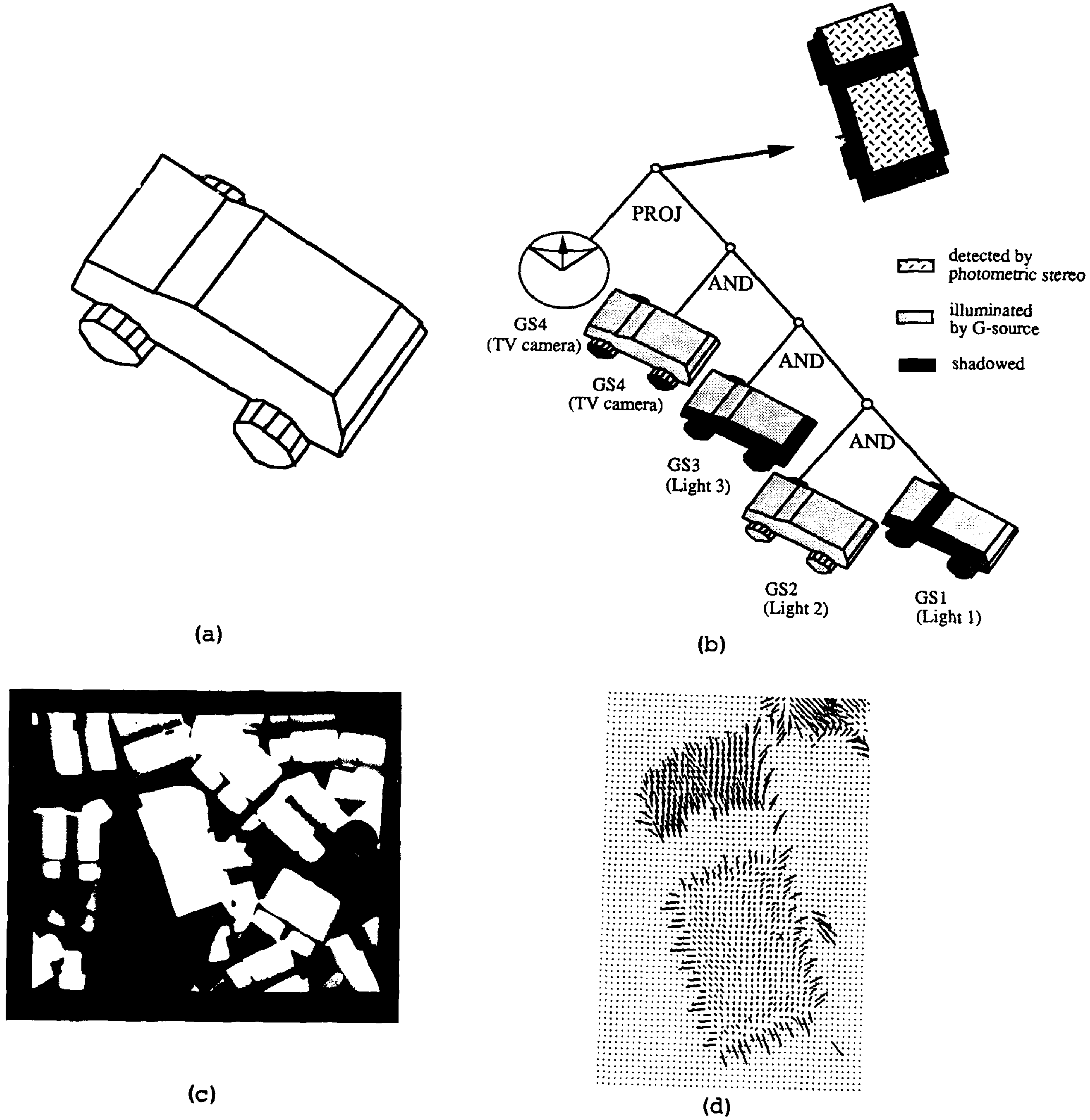


Figure 14: Prcdicted appearance by VANTAGE and obtained appearance: (a) Car model; (b) Applying the sensor composition tree to the car model; (c) Original scene containing the car; (d) A needle map obtained by the photometric stereo. The predicted appearance is consistent with the obtained appearance.

compiler. Since each 2D appearance is represented symbolically, it is relatively easy to construct such an aspect structure (as shown in Figure 15(b)) based on 2D appearance structures generated from the output of VANTAGE. Predicted ranges of uncertainty of geometric features are determined using the sensor reliability and added to the aspect structures.

Our geometric compiler, a program to compile an object model into a localization program, uses aspects as a basic tool for object localization. It generates a two step program; to classify one appearance of an object into one of the possible aspects, and then to determine the precise attitude and position within that aspect.

The geometric compiler generates aspect classification part of the interpretion tree by performing recursive sub-divisions of possible aspects (more precisely aspect components). These sub-division is performed with examining uncertainty ranges of aspect features along the order of the smaller computational cost of them and determining threshold values [Ike89]. Namely, before generation, the compiler calculates computational costs of available features and sorts the features along their computational costs. Figure 16 depicts an example of computation costs of features. The abscissa denotes the sixteen features, which our current implementation uses, while the ordinate denotes their computational costs.

The actual generation begins creating a node which contains all possible aspect components. After this operation, the following operation will be applied recursively to each node containing a group of aspect components along a set of features. At each node, containing a group of aspect components, the compiler examines whether it can divide the group into several groups by thresholding the uncertainty range of one particular feature of the aspect components. If it can, it stores the feature name at the node, generates subnodes corresponding to subgroups of aspect components, and connects them to the node. It also stores the threshold values to divide the groups at the node. If it cannot, it does nothing. Then it repeats the above process using the next feature along the set of features until each node contains only one aspect components or applicable features is exhausted. If exhausted, the compiler forces to divide the node into single aspect component nodes and stores a special rule, *parallel rule.* [10]

The generated classification strategy is represented as a tree structure, which we refer to as an *interpretation tree*, whose intermediate nodes correspond to classification stages of aspects and store feature kinds and values for classifications. Each leaf node contains one particular aspect component.

Each leaf node of the tree, thus if the tree is applied to a real scene, gives a correspondence between an image region and an aspect component (and thus a model face). The compiler generates attitude determination by using these correspondences [IH89]. The compiler has rules concerning how to define a face coordinate system on a model face. Using the rules, the compiler defines a face feature coordinate system on a model face, stores the method in nodes of the tree. The compiler also calculates the transformation from the face coordinate system to the body coordinate system using the geometric model of the object and store it in the nodes of the tree. Thus, after this operation, the object model is represented as a collection of feature coordinate systems and their relationship to the body coordinate system in the interpretation tree, as shown in Figure 17(a). At each leaf node of the tree, the compiler prepares a verification node to calculate the confidence measure of the obtained body coordinate system by comparing the predicted edges and observed edges.

Once this process generates the localization strategy, it converts the strategy into a program using an object library. An object library is a collection of prototypical objects of the object-oriented programming. An object in the object oriented programming is a processing unit, which can perform some operations and store several internal values in slots. We can define demon functions for each slot, where a demon function will be invoked implicitly when we retrieve/insert a value from/into the slot. We can define methods for each object, where a method function will be invoked explicitly when we send a message, assigned to the method, to the object. We can instantiate an instance object from a prototypical object, where the instance object can inherit slot names, slot values, demon functions, and methods. In the compile mode, the geometric compiler retrieves the definition methods stored at each node previously, and instantiates appropriate objects from the object library, and attaches these instance objects to the interpretation tree. See Figure 17(b).

Figure 18(a) shows the generated program [IH89] from the car model. Nodes having *B* and *L* correspond to those for aspect classification and attitude determination, respectively. The generated program is applied to a scene. The program extracts a feature value, specified at each node, from the region, compares it with the threshold feature values, and classifies the region into one or several possible aspects. The generated program estimates the body coordinate systems at these possible aspects. Comparing the edge distributions generated by VANTAGE from the estimated body coordinate systems and the real edge distribution, the program determines the most likely position and attitude, as shown in Figure 18(b), where the most likely one, generated by VANTAGE, is superimposed over the scene.

[10] In the run mode, all the node connected to the parallel node will be executed. The most likely branch under the parallel node will be chosen using the confidence measures given by the verification nodes described below.

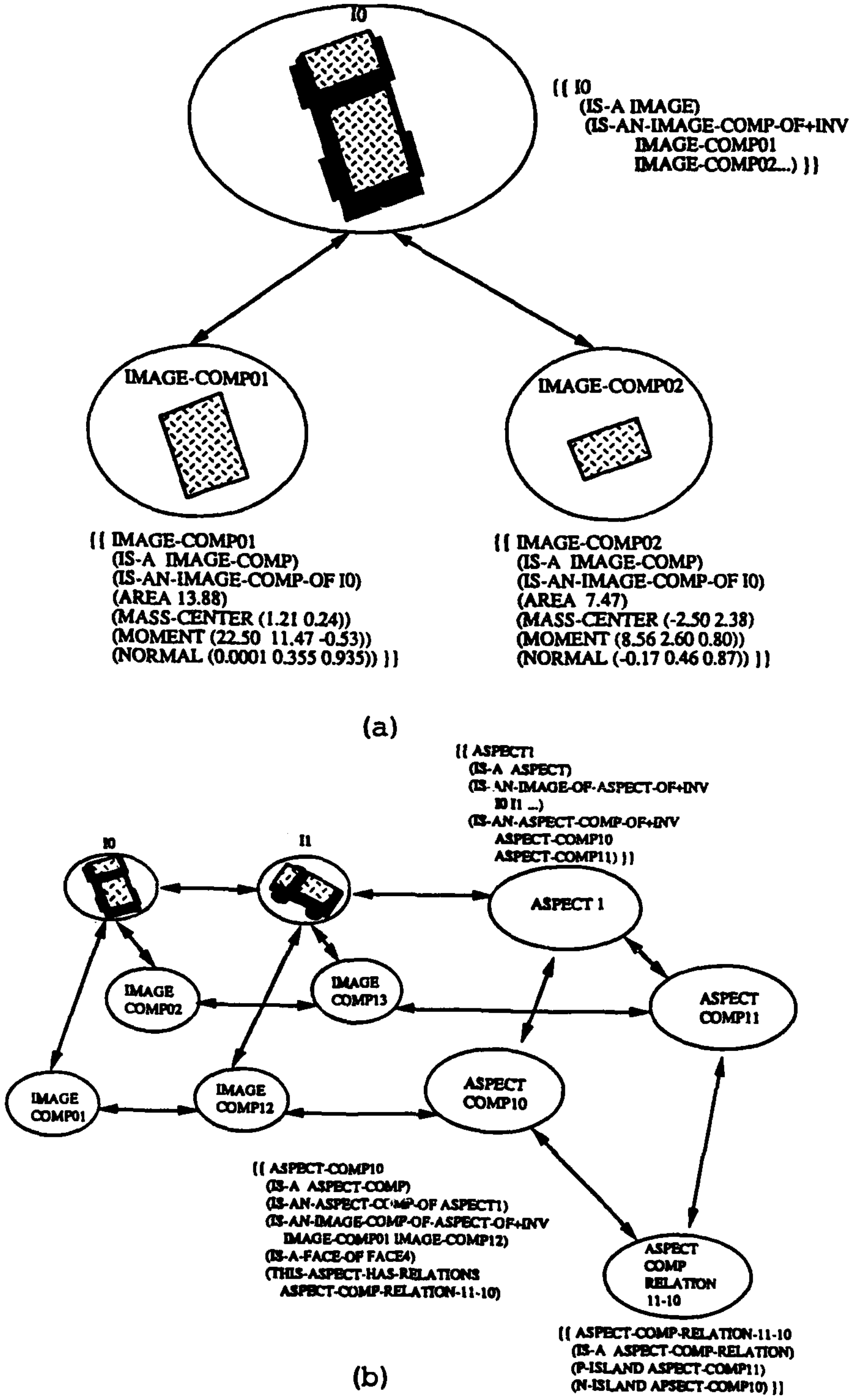
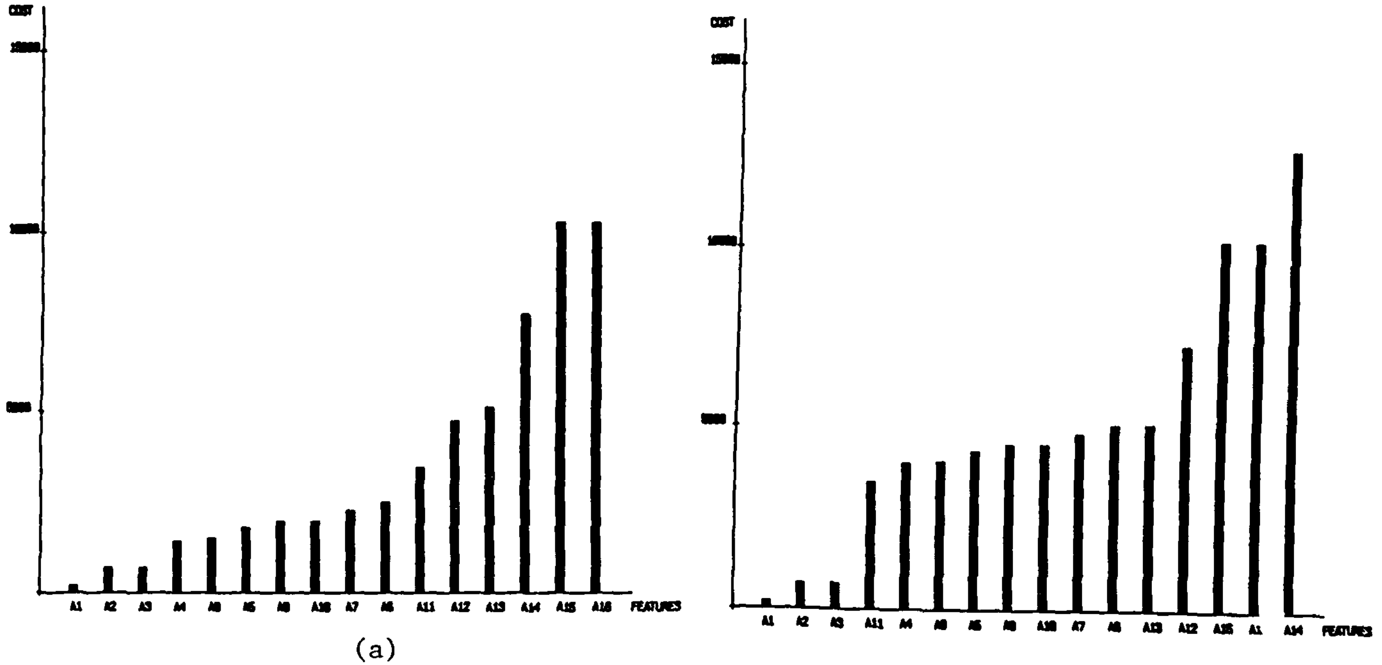

I0
{{ I0
(IS-A IMAGE)
(IS-AN-IMAGE-COMP-OF+INV
IMAGE-COMP01
IMAGE-COMP02...) }}
IMAGE-COMP01
IMAGE-COMP02
{{ IMAGE-COMP01
(IS-A IMAGE-COMP)
(IS-AN-IMAGE-COMP-OF I0)
(AREA 13.88)
(MASS-CENTER (1.21 0.24))
(MOMENT (22.50 11.47 -0.53))
(NORMAL (0.0001 0.355 0.935)) }}
{{ IMAGE-COMP02
(IS-A IMAGE-COMP)
(IS-AN-IMAGE-COMP-OF I0)
(AREA 7.47)
(MASS-CENTER (-2.50 2.38)
(MOMENT (8.56 2.60 0.80))
(NORMAL (-0.17 0.46 0.87)) }}
(a)
{{ ASPECT1
(IS-A ASPECT)
(IS-AN-IMAGE-OF-ASPECT-OF+INV
I0 I1 ...)
(IS-AN-ASPECT-COMP-OF+INV
ASPECT-COMP10
ASPECT-COMP11) }}
I0
I1
ASPECT 1
IMAGE COMP02
IMAGE COMP13
ASPECT COMP11
IMAGE COMP01
IMAGE COMP12
ASPECT COMP10
{{ ASPECT-COMP10
(IS-A ASPECT-COMP)
(IS-AN-ASPECT-COMP-OF ASPECT1)
(IS-AN-IMAGE-COMP-OF-ASPECT-OF+INV
IMAGE-COMP01 IMAGE-COMP12)
(IS-A-FACE-OF FACE4)
(THIS-ASPECT-HAS-RELATIONS
ASPECT-COMP-RELATION-11-10)
ASPECT COMP RELATION 11-10
{{ ASPECT-COMP-RELATION-11-10
(IS-A ASPECT-COMP-RELATION)
(P-ISLAND ASPECT-COMP11)
(N-ISLAND APSECT-COMP10) }}
(b)

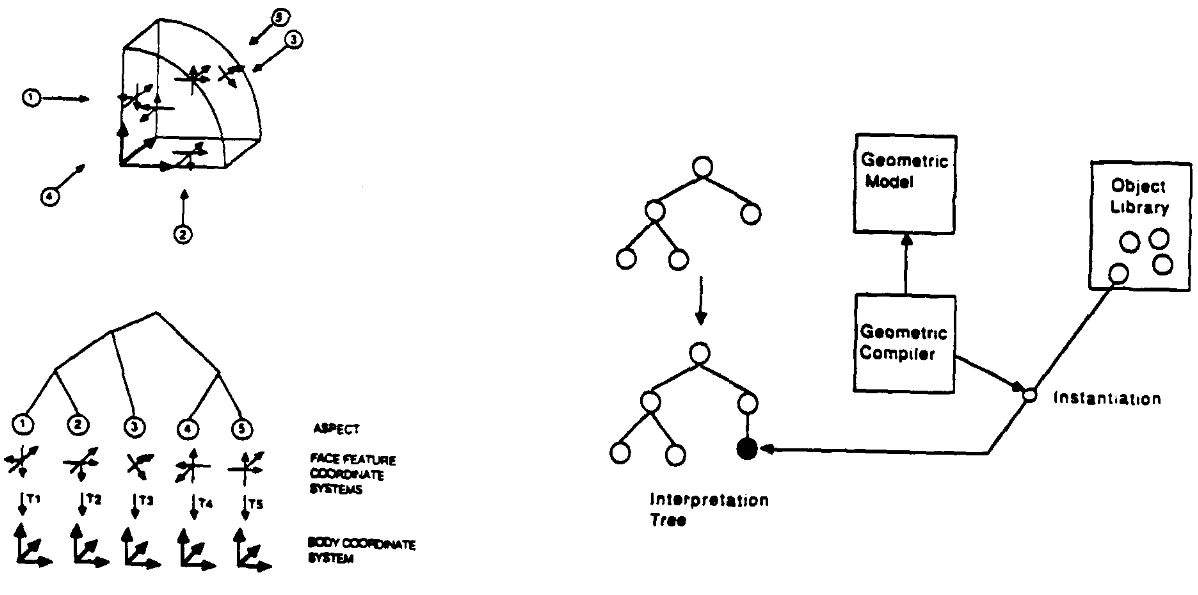
Figure 15: Aspect structure: (a) Image structure. This structure is generated based on the detectable portions, represented using the illumination oriented structure; (b) Aspect structure.



(a)

(b)

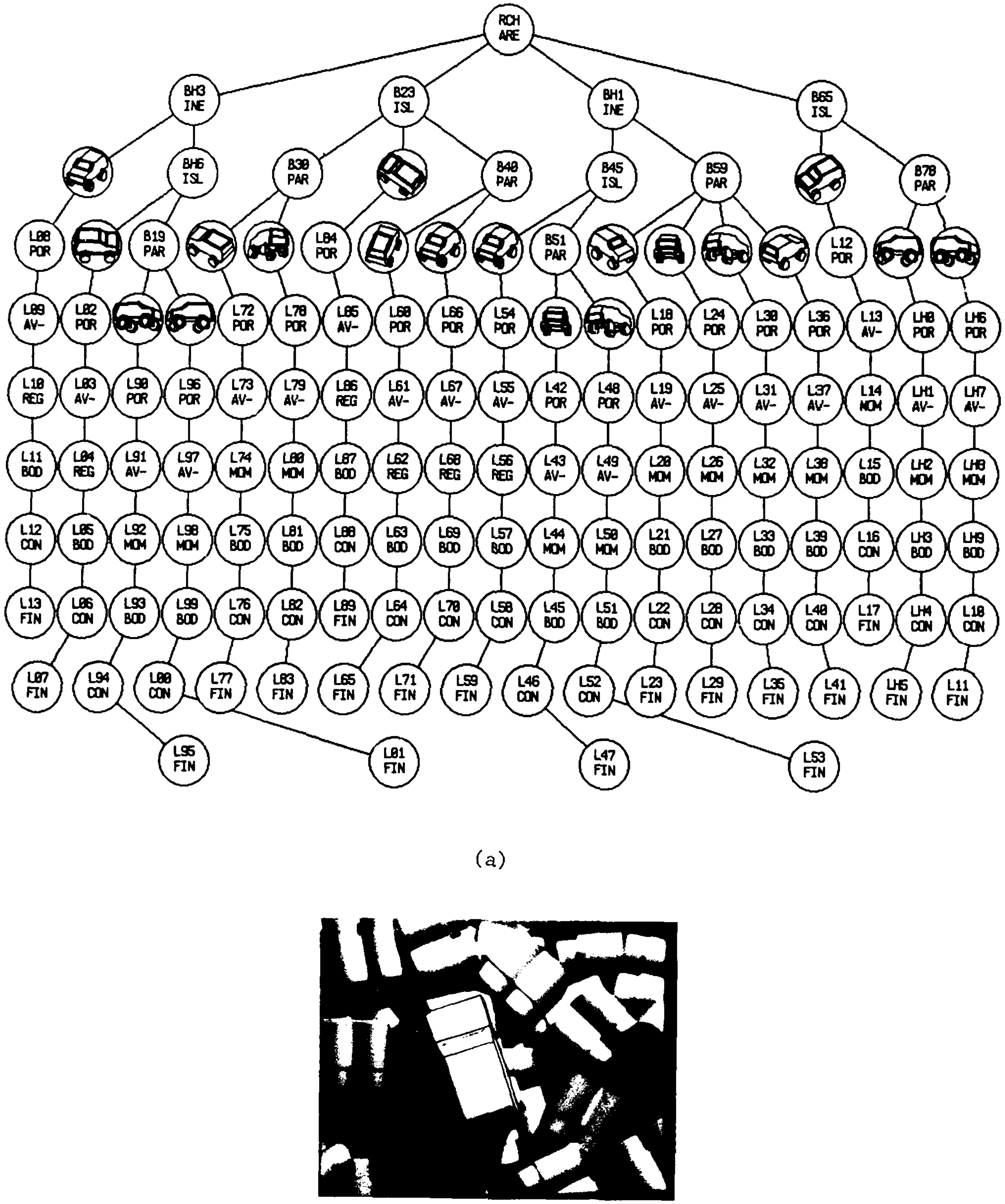
Figure 16: Computational costs of features: The abscissa denotes the order of the sixteen features, which our current implementation uses, while the ordinate denotes their computational costs. The order depends on various factors such as the implementation of the hardware, size of image, and computational method. (a) The cost order in our current implementation; (b) The cost order in the same parameters except the search area of the neighboring region is two times larger.



(a)

(b)

Figure 17: The geometric compiler; (a) The object model is represented as a collection of feature coordinate systems and their relationship to the body coordinate system in the interpretation tree. (b) The geometric compiler uses an object library to convert a recognition strategy into a recognition program.



(a)

(b)

Figure 18: Example; (a) The generated program. The nodes, begins with *B*, correspond to those for aspect classifications and those, begin with*L*, correspond to linear change determination. (b)Predicted object configuration (black lines), generated by VANTAGE and superimposed on the image.

SUMMARY

This paper has described a method of modeling sensor detectability and its implementation in VANTAGE sensor modeler. Each sensor consists of one or more light sources and TV cameras. Sensor detectability, that is where the sensor can detect, can be specified by a set operation of portions illuminated by its light sources and portions visible to its TV cameras.

In order to represent this sensor detectability, we defined a G-source as an abstract sensor component representing either a light source or a TV camera. Then, we proposed G-source illumination conditions to describe under what condition each G-source illuminates surface patches with respect to its illumination direction.

One of the most important findings is that the detectability of a sensor can be decomposed into the sensor's component G-sources' illumination conditions and set operations between them. We can apply its G-sources' illumination conditions independently to an object model, and then apply set operations between them, to obtain detectable portions.

According to these findings, we proposed a sensoı composition tree, which consists of G-source illumination conditions as its leaf nodes and set operations as its branch nodes. These set operations perform the combination of the component G-source illumination conditions. The execution of the sensor composition tree is implemented in three steps: applying the component G-sources' illumination conditions to all 3D faces of the object and storing them as 3D illumination properties on those faces; applying set operations between stored illumination properties; and projecting the resulting illuminated portions onto the image plane to generate a 2D appearance.

We also proposed several 2D structures to represent detected 2D appearances, given by VANTAGE using a sensor composition tree. Since the 3D structure of VANTAGE has four levels of representation: solid, merged, grouped, and approximation and since 2D appearances of 3D scenes can be obtained by projection, illumination information can be stored at, above, between, or below these levels in 2D.

Finally, we showed how to use these capabilities in model-based vision.

Acknowledgments

Takeo Kanade provided many useful comments and encouragements. Keith Petersen proofread earlier drafts of this paper and provided many useful comments which have improved the readability of this paper. Purushothaman Balakumar and Regis Hoffman also partly implemented VANTAGE. Ki Sang Hong actively used VANTAGE for his project and provided many useful comments which have improved the flexibility and stability of this system.

Appendix A: Definition of Coordinate systems

Face We define the z-axis of the feature coordinate system to agree with the surface normal, and the $x - y$axes lie on the face, but are defined arbitrarily otherwise.

Edge We define the z-axis to agree with the average direction of the two normals of the two adjacent faces incident to the edge. We define the x-axis of the feature coordinate system to agree with the edge direction. The y-axis is determined according to x and z.

Vertex We define the z-axis to agree with the average direction of the normals of the adjacent faces incident to the vertex. The x and y axes lie on the plane perpendicular to the z-axis, but are defined arbitrarily otherwise.

References

[Bau72] B.G. Baumgart. *Winged edge polyhedron representation.* Technical Report STAN-CS-320, Stanford University, Artificial Intelligence Laboratory, 1972.

[Bes88] P.J. Besl. Geometric modeling and computer vision. *Proc. of the IEEE*, 76(8):936–958, August 1988.

[Bez72] P. Bezier. *Numerical control: mathematics and applications.* John Wiley, New York, 1972.

[BG82] J.W. Boyes and J.E. Gilchrist. GMSolid: interactive modeling for design and analysis of solids. *IEEE Journal of Computer Graphics and Applications*, 2(2):27–40, March 1982.

[BH86] R.C. Bolles and P. Horaud. 3DPO: a three-dimensional part orientation system. *The International Journal of Robotics Research*, 5(3):3–26, 1986.

[BRH*89] P. Balakumar, J.C. Robert, R. Hoffman, K. Ikeuchi, and T. Kanade. *VANTAGE: A Frame-Based Geometric/Sensor Modeling System – Programmer/User's Manual V1.0.* Technical Report, Carnegie Mellon University, Robotics Institute, 1989. (In preparation).

[HH87] C. Hansen and T. Henderson. CAGD-based computer vision. In *Proc. IEEE Computer Society Workshop on Computer Vision*, pages 100–105, IEEE Computer Society, Miami Beach, FL, December 1987.

[IH89] K. Ikeuchi and K. S. Hong. Determining linear shape change: toward automatic generation of object recognition program. In *Proc. of IEEE Conf. on Computer Vision and Pattern Recognition*, San Diego, June 1989. a longer version, containing programs, is avaiable as CMU-CS-88-188.

[IK89] K. Ikeuchi and T. Kanade. Modeling sensors: toward automatic generation of object recognition program. *Computer Vision, Graphics, and Image Processing*, 1989. (Accepted for publication).

[Ike87] K. Ikeuchi. Generating an interpretation tree from a CAD model for 3-D object recognition in bin-picking tasks. *International Journal of Computer Vision*, 1(2):145–165, 1987.

[Ike89] K. Ikeuchi. A geometric compiler: compiling a geometric model to an object recognition strategy and a grasping strategy for a bin-picking task. *Artificial Intelligence*, 1989. (Submitted).

[IR89] K. Ikeuchi and J. C. Robert. *Modeling Sensor Detectability with VANTAGE Geometric/Sensor Modeler.* Technical Report CMU-CS-89-120, Carnegie Mellon Univeristy, School of Computer Science, February 1989.

[KBR*88] T. Kanade, P. Balakumar, J.C. Robert, R. Hoffman, and K. Ikeuchi. Overview of geometric/sensor modeler VANTAGE. In *Proc. the International Symposium and Exposition on Robots*, The Australian Robot Association, Sydney, Australia, November 1988.

[KH77] F. Kimura and M. Hosaka. *Program Package GEOMAP Reference Manual.* Computer Vision Section, Electrotechnical Lab., 1977.

[Kos84] K. Koshikawa. *SOLVER reference manual.* Computer Vision Section, Electrotechnical Lab., RM-85-33J edition, 1984. (In Japanese).

[KS85] K. Koshikawa and Y. Shirai. A 3-D modeler for vision research. In *Proc. Intern. Conf. on Advanced Robot (ICAR85)*, pages 185–190, Robotics Society of Japan, 1985.

[Loz82] T. Lozano-Perez. Task planning. In M. Brady, J.M. Hollerbach, T.L. Johnson, T. Lozano-Perez, and M.T. Mason, editors, *Robot motion*, chapter 9, pages 473–498, The MIT Press, Cambridge, MA, 1982.

[Mor85] M.E. Mortenson. *Geometric Modeling.* John Wiley, 1985.

[Req80] A. A. G. Requicha. Representations for rigid solids: theory, methods, and systems. *ACM Computing Surveys*, 12(4):437–464, December 1980.

[Rob65] L.G. Roberts. Machine perception of three-dimensional solids. In J.T. Tipplett, editor, *Optical and Electro-Optical Information Processing*, pages 159–197, MIT Press, Cambridge, MA, 1965.

[RV82] A.A.G. Requicha and H.B. Voelcker. Solid modeling: a historical summary and contemporary assessment. *IEEE Computer Graphics and Applications*, 9–24, march 1982.

[SM86] A. Storr and J.F. McWaters. *Off-line programming of industrial robots.* North-Holland, Amsterdam, 1986.

[Sut63] I.E. Sutherland. Sketchpad: a man-machine graphical communication system. In *Proceedings of the Spring Joint Computer Conference: 23*, pages 329–349, 1963.

[WA77] K. Weiler and P. Atherton. Hidden surface removal using polygon area sorting. In *Proc. of SIGGRAPH*, 1977.

INERTIAL NAVIGATION SENSOR INTEGRATED MOTION ANALYSIS

Bir Bhanu, Barry Roberts, and John Ming

Honeywell Systems and Research Center
3660 Technology Drive
Minneapolis, MN 55418

ABSTRACT

Land navigation requires a vehicle to steer clear of trees, rocks, and man-made obstacles in the vehicle's path while vehicles in flight, such as rotorcraft, must avoid antennas, towers, poles, fences, tree branches, and wires strung across the flight path. Automatic detection of these obstacles and the necessary guidance and control actions triggered by such detection would facilitate autonomous vehicle navigation. An approach employing a passive sensor for mobility and navigation is generally preferred in practical applications of these robotic vehicles. Motion analysis of imagery obtained during vehicle travel can be used to generate range measurements to world points within the field of view of the sensor, which can then be used to provide obstacle detection. But, state-of-the-art in motion analysis is not robust and reliable enough to handle arbitrary image motion caused by vehicle movement. However, many types of existing vehicles contain inertial navigation systems (INS) which can be utilized to greatly improve the performance of motion analysis techniques and make them useful for practical military and civilian applications. In particular, INS measurements can improve interest point selection, matching of the interest points, and the subsequent motion detection, tracking, and obstacle detection. We review various techniques of ranging (both passive and active) and discuss an inertial sensor integrated optical flow technique for motion analysis to achieve increased effectiveness in obstacle detection during vehicle motion. Our approach to motion analysis for obstacle detection is illustrated by simulated results and the results obtained using land vehicle data.

1. INTRODUCTION

A desired obstacle detection system for many practical applications should exhibit robustness and should not place unduly excessive size, power, or weight demands on the host vehicle. It should work in day/night/adverse weather conditions and should preferably be covert to minimize the threat to the vehicle and the pilot. The technique used for obstacle detection must also have graceful degradation, instead of total failure, under conditions of limited operability. In recent years, considerable effort has been put toward the detection of obstacles that present themselves primarily to ground vehicles. Using mainly active sensors, such as a laser scanner, obstacles (like fence posts, rocks, vegetation) are detected within the field of view of the vehicle's sensor.[14,37] Other active sensors such as Millimeter Wave[1] (MMW) can detect obstacles such as wires, but the constant and continuous image of these active sensors betrays vehicle's covertness.

Passive sensors, such as a TV camera, are also being used to detect obstacles for ground vehicles.[9,16,27] However, state-of-the-art motion analysis techniques for obstacle detection are not robust and reliable enough for many practical applications. Many of these techniques require that unrealistic constraints be placed on the input data in order to make them work. The largest sources of error are sensor motion and incomplete/ambiguous information in the sensed image data. However, many types of land and air vehicles (e.g. helicopters and military ground vehicles) contain an Inertial Navigation System (INS) whose output can be used for applications beyond the original intent of the system. Such vehicles can use the INS information to greatly simplify some of the functionalities normally provided by computer vision, such as obstacle detection, target motion detection, target tracking, stereo, etc. Figure 1 shows several inertial sensor assembly/packages currently available. In this paper, we make use of INS measurements to enhance the quality and robustness of motion analysis techniques for obstacle detection and thereby provide vehicles with new functionality and capability.

The objective of the work presented in this paper is to describe our maximally passive approach to obstacle detection and to discuss the details of our inertial sensor integrated optical flow analysis technique. In Section 2, we review the principal techniques of passive and active ranging. Section 3 discusses the technical problems associated with motion analysis for passive ranging. We present our new approach to motion analysis in Section 4 and describe the details of the optical flow technique. Section 5 describes the results we have obtained with our optical flow approach. Finally, Section 6 provides the conclusions of the paper. A brief discussion on navigation errors for some ring laser gyros is provided in Appendix A.

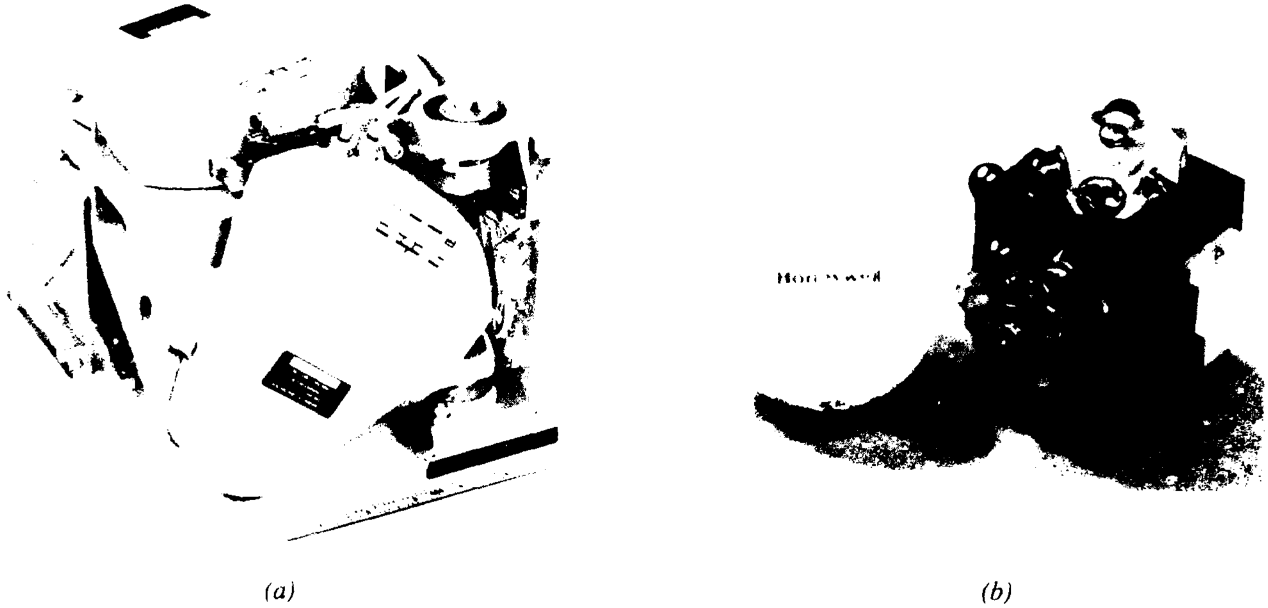


Figure 1: Inertial sensor assemblies. *(a)* The ring laser gyros (RLGs) used in this unit are Honeywell Model GG1342's, which are 4.2 inches in length. This model is the mainline aircraft navigation-grade laser gyro with a performance in the 0.007 degree/hr range. *(b)* Integrated inertial navigation package, which uses Honeywell's GG1308 ring laser gyros, currently under development is extremely compact and can be mounted on a camera.

2. OVERVIEW OF PASSIVE AND ACTIVE RANGING TECHNIQUES

2.1 PASSIVE RANGING

There are a large number of obstacle detection techniques proposed in the literature that make use of motion analysis, stereo methods, or other techniques for passive ranging.

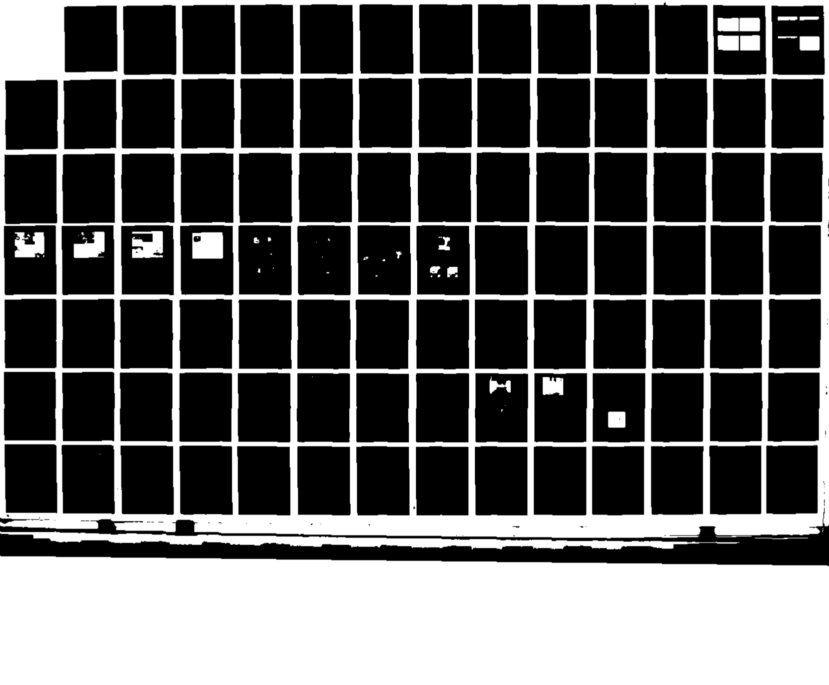
Motion Analysis Techniques - These methods can be further broken down into optical flow approaches or structure and motion methods.

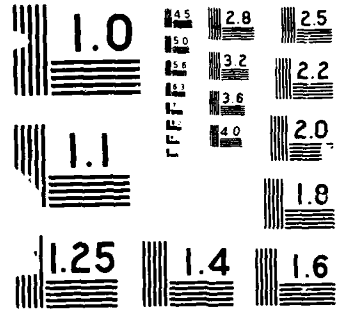
Optical Flow Techniques - These techniques utilize information provided by the velocity field that represents the apparent motion of stationary object points through a temporal sequence of images. Most methods for estimation of optical flow can be categorized into two classes, gradient-based methods and displacement-based methods.

Gradient-Based Methods - These methods use relationships involving optical flow and derivatives of the image brightness function coupled with a variety of constraints on the flow field.[20,22] Although considerable work has been done in this area, significant results have yet to be demonstrated on images of outdoor scenes. One fundamental reason for this is that these methods are highly sensitive to noise. This is a highly undesirable property of any method used on images of outdoor scenes. Furthermore, theoretical analysis has shown that there is a direct conflict between various constraints imposed on the flow field.[23] In particular, it is shown that errors due to instability of solutions of the required systems of equations are inversely related to the size of the neighborhood used for flow smoothness constraints but that increasing the size of the neighborhood increases the error due to violations of flow smoothness.

Displacement-Based Methods - In these methods image features (points, edges, regions, or boundaries) are matched between a temporal sequence of two or more images to derive initial estimates of flow vectors.[5,8,32,35] These methods make use of the flow pattern which is experienced by a moving observer.[24,25,29,30] The motion between frames can be decomposed into translational and rotational components. The final (usually the second) image in the sequence can then be derotated to achieve a relationship approximating pure translation between the first and second image. In the case of pure sensor translation in a stationary environment, every point seems to expand from one particular image location termed the *Focus of Expansion* (FOE). If the location of the FOE is

AD-A212 981 PROCEEDINGS OF IMAGE UNDERSTANDING WORKSHOP HELD IN 9/13
PALO ALTO CALIFORNIA O (U) SCIENCE APPLICATIONS
INTERNATIONAL CORP MCLEAN VA L S BAUMAN MAY 89
UNCLASSIFIED N00014-86-C-0700 F/G 17/11 NL




1.0
4.5
5.0
5.6
6.3
2.8
3.2
3.6
4.0
2.5
2.2
2.0
1.1
1.8
1.25
1.4
1.6

known, then the relative depth of matched stationary image points can be found. If, in addition, the velocity of the sensor and the elapsed time between frames are known, then the absolute range can be computed using trigonometric formulas. Range to other image points can be estimated using interpolation procedures. Accurate range estimates utilizing the above approach require long displacement vectors. Issues raised here include accurate frame-to-frame correspondence, accurate FOE location, and the magnitude of interpolation ambiguities. Inertial navigation sensor (INS) integrated methods presented later in this paper alleviate these problems.

Structure and Motion Methods -- In these methods, both 3-D structure and motion are computed in one integral step by solving a system of linear or nonlinear equations.[17,28,33] These methods, although elegant, reportedly are sensitive to noise, require large amounts of computation, converge slowly, and require many disparate views of the object.

Stereo Techniques - These methods are widely studied for determining range passively.[4,26] In order to use stereo, feature matches must be made between the two images. The accuracy of these matches depends upon the knowledge of relative sensor positions, the displacement between the sensors (the baseline), and the availability of prominent features to match. Once feature correspondences in the two images have been established, the range to the corresponding world points can be computed using trigonometry. Range to other points can then be estimated by using an interpolation procedure. One characteristic property of stereo methods is the fact that, to a first approximation, the error in stereo depth measurements is directly proportional to the positional error of the matches and inversely proportional to the length of the baseline.[4] Thus, as in the case of gradient based techniques, there are sources of error here which are in direct conflict with one another since the longer the baseline is, the harder it is to obtain accurate matches. Potential solutions to this problem include development of highly accurate feature matching techniques and statistical averaging over several views. The statistical averaging method uses the concept of combining motion and stereo, which has the advantage of providing complementary and cooperative information to a passive ranging system.

Other Techniques - Although the aforementioned techniques comprise the majority of methods used in passive ranging, various other approaches have also been suggested.[21] A spatio-temporal extension of the Marr-Hildreth edge operator is one method which has been suggested by Buxton and Buxton.[12] The operator is used to locate edges in time varying imagery. This technique can be considered as a type of hybrid between the gradient and displacement methods. This technique suffers from certain disadvantages such as the Aperture Effect[18] as compared to the wavefront region growing techniques developed by Bhanu and Burger.[7] Hollister has developed a technique for passive ranging to point sources using the bearing angles between the sensor line of sight and the point source.[19] This technique is based on the assumption that all motion is in a plane and no results on real data are given. Bowman and Gross have also developed a method for passive ranging to targets using data from two different aircrafts,[11] but it is not applicable to the rotorcraft low-altitude flight scenario. Techniques based on Kalman filtering are also being developed for general motion and passive ranging.[36]

2.2 ACTIVE RANGING

A variety of laser and Millimeter Wave (MMW) radar systems exist that can scan relatively large fields-of-view and detect and determine the range to power lines, cables, and terrain obstacles. Currently, a number of 3-D laser scanners using phase detection technology are available.[6,31] One such sensor, developed for autonomous vehicle navigation, has a field-of-view of ±40° horizontally and covers depression angles from 15° to 45° with a range resolution of 8 centimeters. More advanced systems with multiple lasers operating at multiple frequencies in the visible, near infrared, and shortwave infrared wavelengths are also under development. The multiple wavelengths allow for range and reflectance determination with a 60° x 80° field-of-view and a range resolution of 2 centimeters. Another commercially available 3-D laser ranging system with a 60° x 60° field-of-view has similar range resolution. It allows a fixed pattern radar scan with 128 x 128 pixel resolution and a frame rate of 0.8 seconds. Both of the above mentioned systems have phase ambiguities on the order of 40 feet in their range measurements.

Thomson CSF[1] is developing a compact MMW radar system (Romeo 2) which uses a 3-second scan over a 90° sector to detect hazardous objects. Prototype systems have detected 3 millimeter diameter high tension cables at ranges of 1000 meters in foggy weather. The system is designed to detect similar objects with small cross sections.

In general, lasers and MMW radar systems are able to detect and accurately determine the range of terrain obstacles. Studies have demonstrated that both kinds of systems can successfully detect transmission cables at all angles, polarizations, and surface conditions, although transmission line detection at all aspect angles with a MMW sensor requires scanning.[3] This study was conducted using four MMW frequencies (18, 34, 56, and 94 GHz), two laser wavelengths (10.6, 1.06 μm), three polarizations (horizontal, vertical, and cross), various surface conditions (dry, wet, rough, and smooth), five kinds of cables, and several aspect angles. Recent advances in CO_2 laser technology have led to the development of fieldable LADAR system that can provide high resolution imagery suitable for automatic target recognition and obstacle detection such as wires, poles, etc.

Unfortunately, active systems are subject to threats through automatic detection by the enemy. They undoubtedly provide good obstacle avoidance capability at the price of increased danger to the crew and the vehicle, regardless of whether the active system is based on laser or MMW radar ranging. Consequently, their use should be contingent on the capabilities of passive obstacle detection/avoidance technology in near and far future systems. The report by Bhanu and Roberts[9] presents detailed tradeoffs of passive/active ranging approaches for obstacle detection.

3. TECHNICAL PROBLEMS WITH MOTION ANALYSIS TECHNIQUES

In this section, we present a critical assessment of some of the problems in the area of motion analysis that demand innovative solution concepts for success. The difficulties involved come from many sources. The general problem areas which we consider important are (not necessarily in order of importance):

Inherent Quantization Error and Noise -- As mentioned earlier, techniques based on discrete differentiation (optical flow with global constraints[20]) are generally considered to be so sensitive to noise that they are unreliable in outdoor scenes. In addition, the identification of the same world point (interest point) within multiple frames becomes unreliable due to noise and quantization.

Correspondence or Feature Matching Problem -- This problem has been studied quite extensively.[5,10] Solution of this problem is necessary in a purely passive, image-based displacement method as well as for stereo techniques. The great deal of effort which has been expended toward solving this problem and the lack of a technique that will insure a very high degree of matching accuracy implies that passive ranging systems which utilize feature matching must be capable of tolerating a certain number of inaccurate (possibly highly inaccurate) matches. One well known approach to increasing the accuracy of matches obtained by the correspondence problem is to use relaxation techniques to maintain certain consistencies in correspondences between neighboring features. Another approach is to use input from a Ring Laser Gyro (RLG)/accelerometer apparatus to determine the motion of the sensor between frames in the monocular case, or the relative motions of the sensors in the stereo case, in order to register the images and locate the FOE without computation. This apparatus can provide all the sensor attitude and velocity information needed to completely describe sensor motion. This information also introduces a number of constraints on the search area required for matching. This approach will be explained in more detail in Section 4.

Determining the Location of the FOE -- This is a major problem which must be addressed if one uses optical flow methods and monocular sequences for passive ranging. Location of the FOE can be approximated by purely image-based methods[8] or by using input from a RLG/accelerometer assembly. When using purely image-based methods, bad correspondences, quantization effects (including roundoff error), and noisy data all contribute to the inherent instability in attempting to solve a set of equations exactly for the FOE. The method of choice is to use the inertial information provided by an RLG/accelerometer to calculate the FOE within each image frame, thereby avoiding the uncertainties involved with passive methods.

Interpolation of the Range Map -- The desire to have range estimates to all points in the field-of-view is another problem which should be addressed in order to develop accurate passive ranging systems. The previous techniques for passive ranging will compute range only as subset of the available points in the field-of-view. In order to solve the obstacle detection/avoidance problem via ranging, a much more dense set of points must have range values available. An interpolation procedure can be used to estimate range over a dense set (possibly all) of pixels that cover the field-of-view. There are some difficulties associated with this process. A standard practice has been to fit a smooth surface, such as a polynomial of two variables or a spline, through the computed range points. Some problems with polynomials are that they are continuous whereas range maps, in general, are discontinuous. Polynomials also have a predetermined shape. To obtain a high degree of variability, one must use fairly high order polynomials, but then the least squares process becomes very computationally intensive. Splines offer a better choice for fitting a smooth function through the points, since, although they are continuous, a high degree of variation can be attained with reliable, less expensive computational methods. Unfortunately, splines are not guaranteed to pass through the given range points. Scene analysis techniques can simplify this process.[9]

Use of Artificial Intelligence and Qualitative Methods -- A final general issue to consider in motion analysis is not a direct problem. The issue is how much, if any, do passive ranging via motion analysis methods need to be augmented with intelligent, or qualitative, techniques. Range is a very concrete concept, and it is easy to understand how and why to use range once it is available. It is true, however, that most humans and animals operate passively with only a very fuzzy sense of absolute range. Motion cues, such as occlusion, a priori knowledge, expectations concerning object sizes and characteristics, and contextual cues may be instrumental in enabling biological entities to detect obstacles while navigating through their environment, and may, in the final analysis, be necessary in order to solve the obstacle detection problem satisfactorily using passive sensors. Encapsulating these types of information, which are based on very abstract concepts, is very difficult and is an active area of research in artificial intelligence and image and motion understanding. The solution of a number of problems in scene analysis and understanding as well as knowledge representation and utilization need to be solved before a reliable method for obstacle detection using a significant amount of knowledge-based reasoning can be developed.

The contents of section 4 describe our approach to motion analysis and describe some of our methods for dealing with the problems listed above.

4. INERTIAL SENSOR INTEGRATED MOTION ANALYSIS

The purpose of this section is to describe the inertial sensor integrated motion analysis approach we have undertaken. The block diagram of this system is illustrated in Figure 2. The system uses inertial sensor integrated optical flow, scene analysis, and selective applications of active sensors to provide obstacle detection capability.[9] In this paper, we focus on the details of the inertial sensor integrated optical flow algorithm, which computes range to features within the sensor's field of view. For a pair of image frames, the major steps that are involved within the optical flow algorithm are give below:

(1) Input frames, frame A and frame B, are read in along with their associated inertial data.

(2) Interest points are extracted from each of the input frames.

(3) Location of the focus of expansion (FOE) (in both frames) is computed.

(4) FOE and the interest points in frame B are projected onto an image plane that is parallel to the image plane that captured frame A (*derotation* of frame B).

(5) Interest points in frame B are matched to those of frame A based upon four criteria.

(6) Range is computed to each interest point in frame B that has a match in frame A.

(7) A dense range map is created using context dependent scene analysis and interpolating between the computed range values.

Before starting a detailed discussion of the major steps in the algorithm, let us first describe the coordinate systems that are used. The digitized imagery contains pixels addressed by row and column with the origin of the 2-D coordinate system located in the upper left corner of the image. The horizontal axis, c, points to the right and the

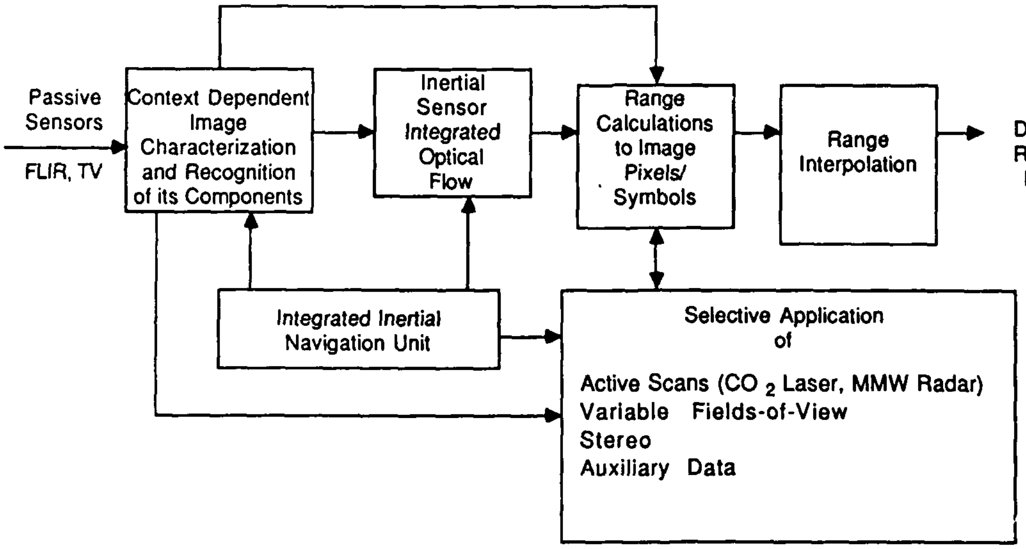

Passive Sensors
FLIR, TV
Context Dependent Image Characterization and Recognition of its Components
Inertial Sensor Integrated Optical Flow
Range Calculations to Image Pixels/ Symbols
Range Interpolation
Dense Range Map
Integrated Inertial Navigation Unit
Selective Application of
Active Scans (CO 2 Laser, MMW Radar)
Variable Fields-of-View
Stereo
Auxiliary Data

Figure 2: Inertial sensor integrated optical flow and scene analysis using both passive and selective applications of active sensors provide robust image analysis useful for obstacle detection/avoidance by a robotic land vehicle or helicopter.

vertical axis, r, is in the downward direction. This image plane is perpendicular to the x axis of a 3-D coordinate system and is located at a distance of the focal length, F, from the origin with the z axis in the downward direction. Therefore, the pixels in the image plane can be described in the 2-D coordinate frame as (c, r) and in the 3-D coordinate frame by the vector (F, y, z). The geometry described above is graphically illustrated in Figure 3.

As shown in Figure 4, the data input to the obstacle detection algorithm consists of a sequence of digitized video or FLIR frames that are accompanied by inertial data consisting of rotational and translational velocities. This information, coupled with the temporal sampling interval between frames, is used to compute the distance vector, $\vec{d}$, between each pair of frames and the roll, pitch and yaw angles, (ϕ,θ,ψ), of each frame. Both $\vec{d}$ and (ϕ,θ,ψ) are crucial to the success of the algorithm described in the following section. There are several other possible variations which we do not discuss in this paper.

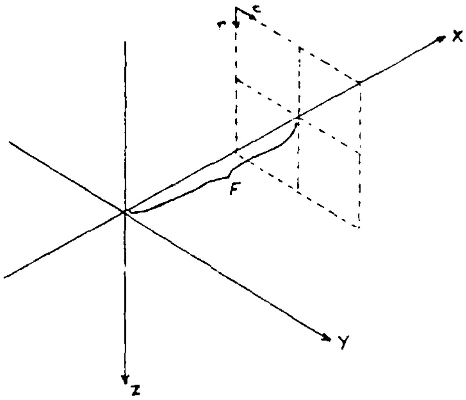


Figure 3: The coordinate system geometry of the sensor's image plane is perpendicular to the x axis, located at the distance of the focal length, F, from the origin of the coordinate system.

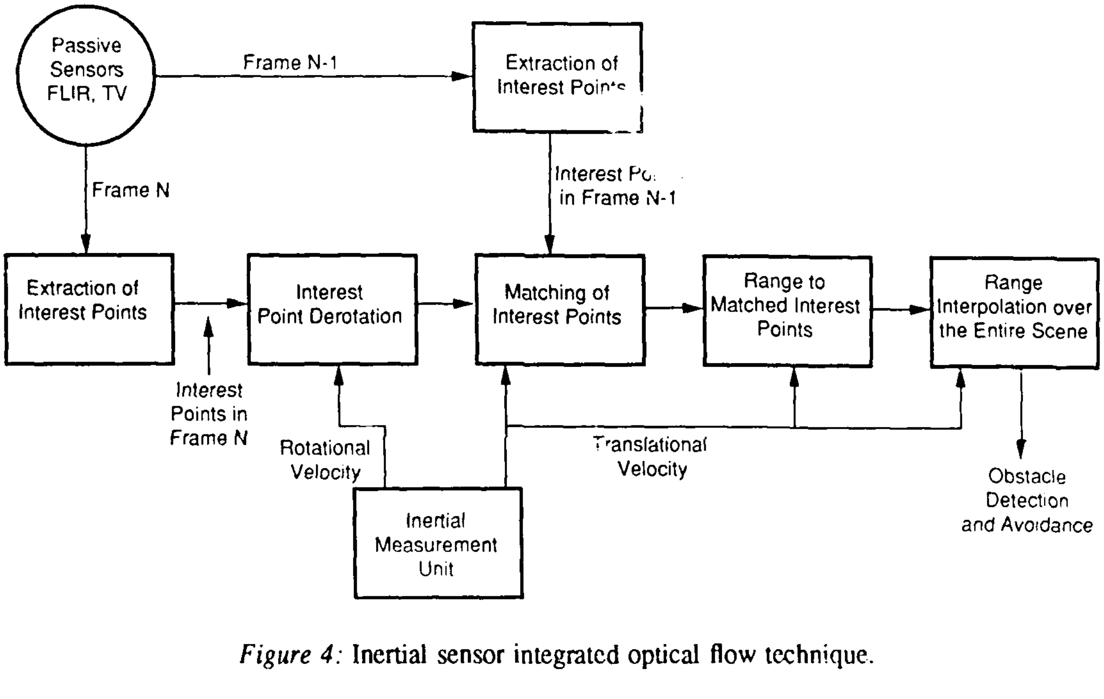

Passive Sensors FLIR, TV
Frame N-1
Extraction of Interest Points
Frame N
Interest Points in Frame N-1
Extraction of Interest Points
Interest Point Derotation
Matching of Interest Points
Range to Matched Interest Points
Range Interpolation over the Entire Scene
Interest Points in Frame N
Rotational Velocity
Translational Velocity
Inertial Measurement Unit
Obstacle Detection and Avoidance

Figure 4: Inertial sensor integrated optical flow technique.

4.1 DISTINGUISHED FEATURES

The features within the imagery (TV or FLIR) that are most prominent and distinguished, mark the world points to which range measurements will be made. These prominent world points, known as *interest points*, are easy to extract from the imagery and have the highest promise of repeated extraction throughout multiple frames. The interest points within the field-of-view of the monocular sensor are of fundamental and critical importance to optical flow calculations. In the following subsections, the extraction and subsequent use of interest points is described in detail.

4.1.1 Interest Point Selection

The computation of distinguishable points is accomplished by passing a Moravec operator[5] over each frame of imagery. The operator is applied to each image pixel (within a desired offset from the image border) which was identified as a strong edge pixel by a Sobel edge operator. The interest operator examines all pixels within a square window, of side length L, that surrounds each edge pixel and computes the relative variance between pixel values. As each pixel within the window is examined, the square of the difference between its value and the values of its neighboring pixels is computed and summed. Actually, four different sums are recorded which correspond to the same 4 neighbors relative to each pixel within the window; there is a sum for the square of the difference between the current pixel and its neighbor to the right and likewise for three other neighbors (below, below & right, below & left). After each pixel under the window has contributed to the 4 sums, the smallest of the sums, S, is selected and stored as the pixel's value. A pixel is deemed an interest point if its assigned value of S is greater than the corresponding sum generated at each pixel within a square window of side length K, centered on the pixel in question. In the discussion that follows, a pixel's value of S will be referred to as its *interestingness.*

Our implementation of the Moravec operator ranks the detected interest points (pixels with a value of S which is a local maximum) in the order of their computed interestingness. This interest point extraction routine divides the image into M uniform regions and returns only the N points within each region which have the highest values of S, where N and M are inputs to the program. The result of returning only the best interest points (in terms of S) in each region is that the processed scene is more uniformly covered with interest points. If this were not the case, a small number of occasionally adjacent regions will lay claim to the major portion of interest points.

Unfortunately, not all regions within a scene can contain reliable interest points (e.g. wave crests on a body of water are not good interest points). Scene analysis techniques are used to ascertain the *goodness* of regions prior to interest point selection.[9] Moreover, interest point selection can be further improved by incorporation of Kalman filtering techniques, which use inertial sensor data to track and predict interesting point features.

4.1.2 Interest Point Derotation

To aid the process of interest point matching, we must make it seem as though image plane B is parallel to image plane A. If this is done, the FOE and pairs of interest points in frames A and B that match, would ideally be colinear should the image planes be superimposed (see Figure 5). To make the image planes parallel, derotation is performed for each vector, (F,y_i,z_i) that corresponds to each interest point in frame B. The equation for the derotation transformation and projection (in homogeneous coordinates) is

$$\begin{bmatrix} F \\ y_i' \\ z_i' \\ 1 \end{bmatrix} = P\, R^{-1}_{\phi_A}\, R^{-1}_{\theta_A}\, R^{-1}_{\psi_A}\, R_{\psi_B}\, R_{\theta_B}\, R_{\phi_B} \begin{bmatrix} F \\ y_i \\ z_i \\ 1 \end{bmatrix} = P\, C^{A}_{NED}\, C^{NED}_{B} \begin{bmatrix} F \\ y_i \\ z_i \\ 1 \end{bmatrix}$$

where

$$R_\phi = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & \cos\phi & \sin\phi & 0 \\ 0 & -sin\phi & \cos\phi & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad R_\theta = \begin{bmatrix} \cos\theta & 0 & -sin\theta & 0 \\ 0 & 1 & 0 & 0 \\ \sin\theta & 0 & \cos\theta & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad R_\psi = \begin{bmatrix} \cos\psi & \sin\psi & 0 & 0 \\ -sin\psi & \cos\psi & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad P = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 1/F & 0 & 0 & 0 \end{bmatrix}$$

and where NED (*north, east, down*) is the coordinate frame in which inertial measurements are made. Use of the NED frame assumes that vehicle motion is "local" to a patch of Earth.

The matrix P projects a world point onto an image plane and is used to compute the FOE, $FOE = P\,\vec{d}$, where $\vec{d} = \vec{v}\Delta t$. The matrix C^{A}_{NED} converts points described in the NED coordinate frame into an equivalent description within a coordinate frame parallel to the A coordinate frame. Likewise, the matrix C^{NED}_{B} converts the descriptions of points in the B coordinate frame into descriptions in a coordinate frame parallel to NED.

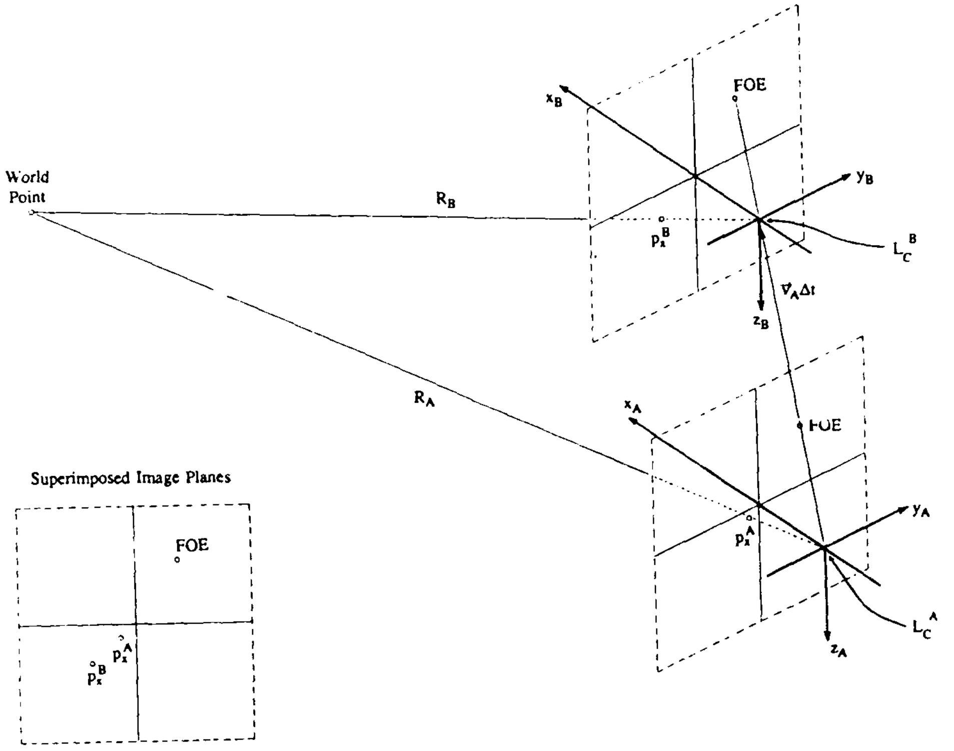


Figure 5: An illustration of the sensor geometry that records two perspective views of a scene at two positions separated by a distance $|\vec{v}|\Delta t = |\vec{d}|$ (with no rotation of the sensor between positions). When there is no rotational change between image frames, there is a special property of the perspective projection of a world point onto the two image planes; the FOE and the projections of the world point are all colinear.

4.1.3 Interest Point Matching

The matching of interest points is performed in two passes. The goal of the first pass is to identify and store the top three candidate matches for each interest point in frame B, (F, y_{B_j}, z_{B_j}). The second pass looks for multiple interest points being matched to a single point in frame A. Hence, the result of the second pass is a *one-to-one* match between the interest points in the two successive frames. For our application, a one-to-one match of interest points is necessary. We acknowledge that the projection onto the sensor's image plane of an object in the world will grow in size as the sensor moves toward the object. This situation might imply that a one-to-one match does not make sense since what was one pixel in size in frame A might become two or more pixels in size in frame B. In this work, we assume that the growth of objects, in terms of pixel size, is negligible in the passive ranging for obstacle detection scenario. All objects are assumed to be at certain safe distances for vehicle maneuvering and one pixel (of interest point quality) in two frames is all that is required of an object's surface for the range to the object to be computed.

Pass One:

To determine the candidate matches to (F, y_{B_j}, z_{B_j}), each of the interest points in frame A is examined with the successive use of four metrics. The *first metric* makes certain that candidate matches lie within a cone shaped region bisected by the line joining the FOE and the interest point in frame B. This metric limits candidate matches to lie within the cone with apex at the FOE, as shown in Figure 6(a). If an interest point in frame A, (F, y_{A_i}, z_{A_i}), passes the first metric, then the second metric is applied to it. The *second metric* requires that the interestingness of candidate matches is close to the interestingness of the point that we are trying to match.

The *third metric* restricts all candidate matches in frame A to lie closer to the FOE than the points in frame B (as physical laws would predict for stationary objects). This metric involves the computation of the distances of the interest points from the FOE, which can be computed in two different ways:

(1) The direct euclidean distance, d_1, between (F,y_{A_i},z_{A_i}) and (F,y_{B_j},z_{B_j}), and

(2) the distance d_2 which is the projection of d_1 onto the line joining (F,y_{B_j},z_{B_j}) and the FOE.

The distance measures are graphically illustrated in Figure 6(b). Regardless of the way that the distance measure is computed, it can be used to identify the closest candidate matches to (F,y_{B_j},z_{B_j}).

The *fourth metric* constrains the distance between an interest point and its candidate matches. For an interest point in frame A, A_j, to be a candidate match to point B_j, it must lie within the shaded region of Figure 6(a). The depth of the region is determined by this fourth metric while the width of the region is fixed by an earlier metric. By limiting interest points, A_i, to lie in the shaded region, we have effectively restricted the computed range of resulting matches to lie between R_{max} and R_{min}. The reasoning behind this restriction is that world objects of range less than R_{min} should not occur due to autonomous or manual navigation of the vehicle, thus avoiding potential collisions. Likewise, objects at a range greater than R_{max} are not yet of concern to the vehicle.

The result of the first pass of interest point matching is a list, for each (F,y_{B_j},z_{B_j}), of three or fewer candidate matches that pass all metrics and have the smallest distance measures of all possible matches.

Pass Two:

The goal of the second pass of the matching process is to take the matches provided by the first pass and generate a *one-to-one* mapping between the interest points in frames A and B. Initially, it can be assumed that the best match to (F,y_{B_j},z_{B_j}) will be the stored candidate match which has the smallest distance measure. Unfortunately, there may be multiple points, (F,y_{B_j},z_{B_j}), which match to a single (F,y_{A_i},z_{A_i}). Hence, the recorded list of best matches is searched for multiple occurrences of any of the interest points in frame A. If multiple interest points in frame B have the same best match, then the point, B^*, which is at the minimum distance from the A_i in question, will retain this match and is removed from the matching process. The remaining B_j's are returned to the matching process for further investigation after having A_i removed from their lists of best matches. This process continues until all of the interest points in frame B either have a match, or are determined to be unmatchable by virtue of an empty candidate match list. Hence, the final result of the matching process is a *one-to-one* mapping between the interest points in frames A and B.

4.2 RANGE CALCULATION AND INTERPLATION

Given the result of interest point matching, which is the optical flow, range can be computed to each match. Given these sparse range measurements, a range or obstacle map can be constructed. The obstacle map can take many forms,[13,34] the simplest of which consists of a display of bearing versus range. In what follows, range calculation is described and the important issue of range interpolation is discussed.

Given pairs of interest point matches between two successive image frames and the translational velocity between frames, it becomes possible to compute the range to the object on which the interest points lie. One approach to range,

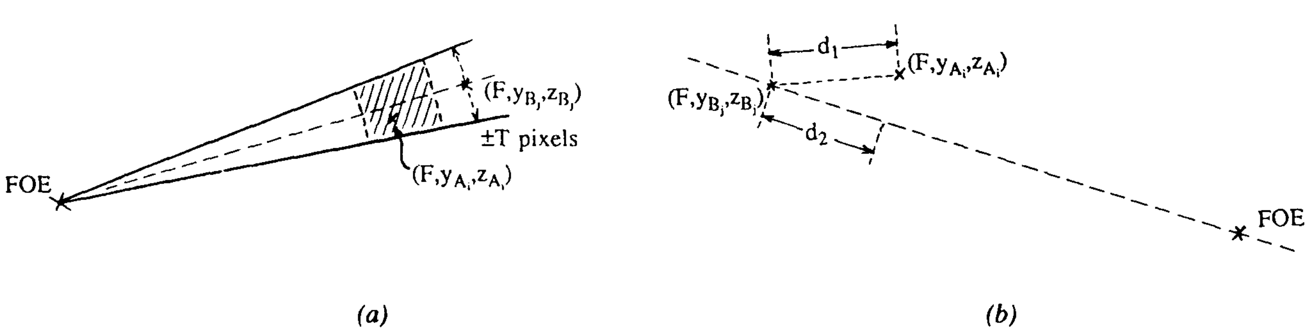


Figure 6: Constraints used to aid the process of matching interest points between frames. *(a)* Since an interest point, the FOE, and a candidate match must be colinear after derotation, all candidate matches to a point in frame B must lie within a cone with apex at the FOE and the shaded section. *(b)* There are two ways to compute the distance between interest points, distance metric d_1 or d_2.

R, computation is described by the equation

$$R = \Delta Z \frac{x' - x_f}{x' - x}$$

where

x_f = the distance between the FOE and the center of the image plane,

x = the distance between the pixel in frame A and the center of the image plane,

x' = the distance between the pixel in frame B and the center of the image plane,

$\Delta Z = |\vec{v}|\Delta t \cos\alpha$ = the distance traversed in one frame time, Δt, as measured along the axis of the line of sight,

α = the angle between the velocity vector and the line of sight,

$x' - x_f$ = the distance in the image plane between (F, y_{B_j}, z_{B_j}) and the FOE, and

$x' - x$ = the distance in the image plane between (F, y_{B_j}, z_{B_j}) and (F, y_{A_i}, z_{A_i}).

These variables are illustrated in Figure 7.

An alternate approach involves the calculation of the angles α_A and α_B between the translational velocity vector and the vectors that describe the matched pair of interest points in frames A and B,

$$R_A = \frac{|\vec{v}|\Delta t \sin\alpha_B}{sin(\alpha_B - \alpha_A)}$$

as indicated in Figure 8. Both of the range calculating techniques compute the distance to a world point relative to the lens center of frame A (similar equations would compute the distance from the lens center of frame B). The accuracy of the range measurements that result from either approach is very sensitive to the accuracy of the matching process as well as the accuracy of the inertial measurement unit (IMU) data.

The task of range interpolation is the last processing step required of the passive ranging system (this ignores any postprocessing of the range that may be required before it gets passed to the automatic vehicle control and display systems). The purpose of this task is to create, by means of interpolation between the sparse range samples generated from the optical flow measurements, a dense range map representing the objects within the field of view. Essentially, this task is one of surface fitting to a sparse, nonuniform set of data points. To obtain an accurate surface fit that

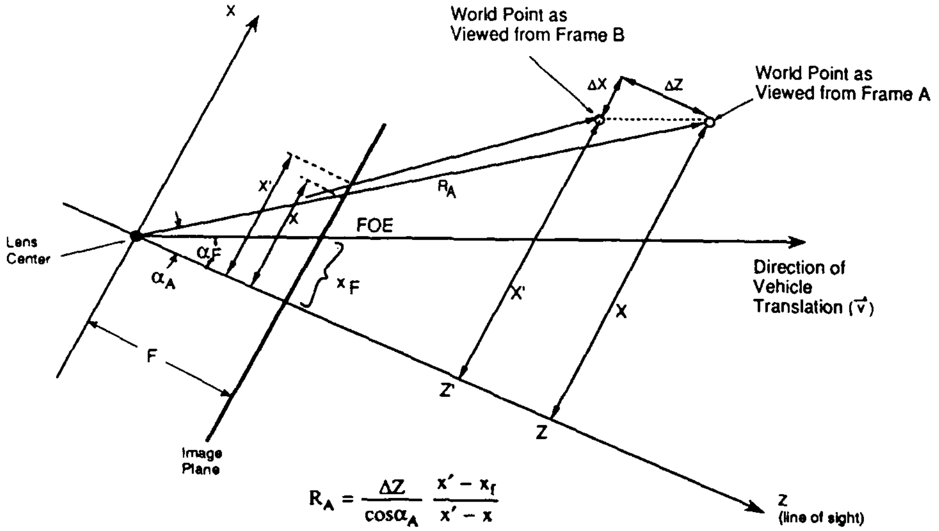
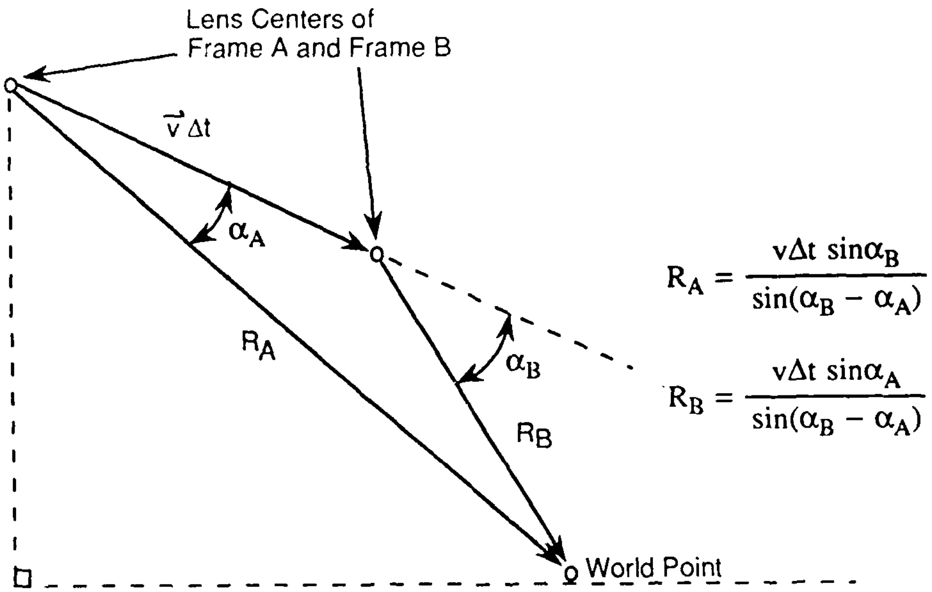

X
World Point as Viewed from Frame B
ΔX
ΔZ
World Point as Viewed from Frame A
x'
x
R_A
FOE
Lens Center
α_F
α_A
x_F
X'
X
Direction of Vehicle Translation ($\vec{v}$)
F
Z'
Z
Image Plane
$R_A = \frac{\Delta Z}{\cos\alpha_A} \frac{x' - x_f}{x' - x}$
Z (line of sight)

Figure 7: The geometry involved in the first approach to range calculation is illustrated here. The figure shows the imaged world point in motion rather than the sensor, thus simplifying the geometry.



$$R_A = \frac{v\Delta t \sin\alpha_B}{\sin(\alpha_B - \alpha_A)}$$

$$R_B = \frac{v\Delta t \sin\alpha_A}{\sin(\alpha_B - \alpha_A)}$$

Figure 8: An alternate approach for range calculation which requires the computation of angles between the linear velocity vector and the vectors that describe the matched pair of interest points.

physically corresponds to the scene within the field of view, it is necessary that the sparse set of range samples be as uniformly spread throughout the field of view as possible. This will require processing steps hinted at in previous sections; scene understanding/segmentation must be used to create regions from which a desired number of interest points are extracted.

The type of surface fitting is important because the resulting surface (i.e. the range map) must pass through each of the range samples. It would be especially dangerous if the surface passed under any range samples. There are many techniques of surface fitting available to our task. To date, we have explored a method of bivariate interpolation over irregular spaced samples proposed by Akimo.[2] This technique uses 5th degree polynomials to interpolate over the triangular regions formed by the range samples. The major drawback associated with this approach is its assumption that all of the given points fall within a convex region. A solution to this problem is to use an improved Delaunay-based triangularization of the range samples, proposed by DeFloriani *et al*,[15] which works over arbitrarily shaped regions of interest.

A less elaborate technique of range interpolation consists of a fitting of planes to the available range samples. This approach gets the job done quickly and efficiently and does succeed in passing through each range sample. All techniques of range interpolation should be careful to avoid interpolation over discontinuities that occur between range samples on the surface under investigation. With the use of scene analysis/segmentation, the smoothing of discontinuities can be avoided by interpolating only over *smooth* regions or segments of the scene. Techniques of joining the regions after interpolation have yet to be developed.

Finally, there is some concern as to the purpose of interpolation. Surely, interpolation will aid an operator/pilot in the interpretation of the results of optical flow measurements, but its use by automatic vehicle control is questionable. Also, a large number of interest points can be selected and matched, so there may not be any need for interpolation. These issues are being explored further.

5. RESULTS

Our inertial navigation sensor integrated optical flow algorithm has been used to generate range samples using both synthetic data and real data (imagery and INS information) obtained from a moving vehicle. In this section, we describe the conditions under which the data was created/collected and provide images illustrating the results of the major steps in the optical flow algorithm.

The synthetic interest points were generated from a file containing the 3-D coordinates of 15 world points. Table 1 shows the 3-D locations of these world points. In the same coordinate system as the interest points are located, Table 2 lists the location, roll, pitch, and yaw of the camera at the two instances of time at which frames A and B were acquired. The time between frame acquisition is 0.2 seconds. Figure 9(a) shows the locations (circles) of the projection of the world points onto the first location of the image plane where the field of view of the synthesized camera model is 52.0° x 48.75° with a focal length of 9 mm. Figure 9(b) shows the locations (squares) of the projections of the world points onto the second location of the image plane and shows the new locations (diamonds) of those projections after derotation. Figure 9(c) shows the results of the matching process in which circles are connected to their corresponding diamond with a straight line and the FOE is labeled and marked with an X. The final frame, Figure 9(d), shows the computed range to each point resulting from each of the matches.

A pair of real images was selected to test the capabilities of the optical flow algorithm using real imagery. Table 3 indicates the location, roll, pitch, and yaw of the camera associated with the pair of real image frames that were used. The field of view of the camera for the real images is 52.1° x 40.3° and the focal length = 9 mm. The elapsed time between the two frames for this experiment was 0.2 seconds. Figure 10(a) shows the locations of the extracted interest points obtained from the first frame, drawn as circles. Similarly, Figure 10(b) indicates the location of extracted interest points (squares) and the corresponding derotated locations (diamonds). Since the vehicle undergoes very little rotation between frames, the derotated locations are nearly coincident with the original point locations. The results of the point matching process for the real imagery is shown in Figure 10(c). Finally, the computed range to each of the matched points is displayed in Figure 10(d).

6. CONCLUSIONS

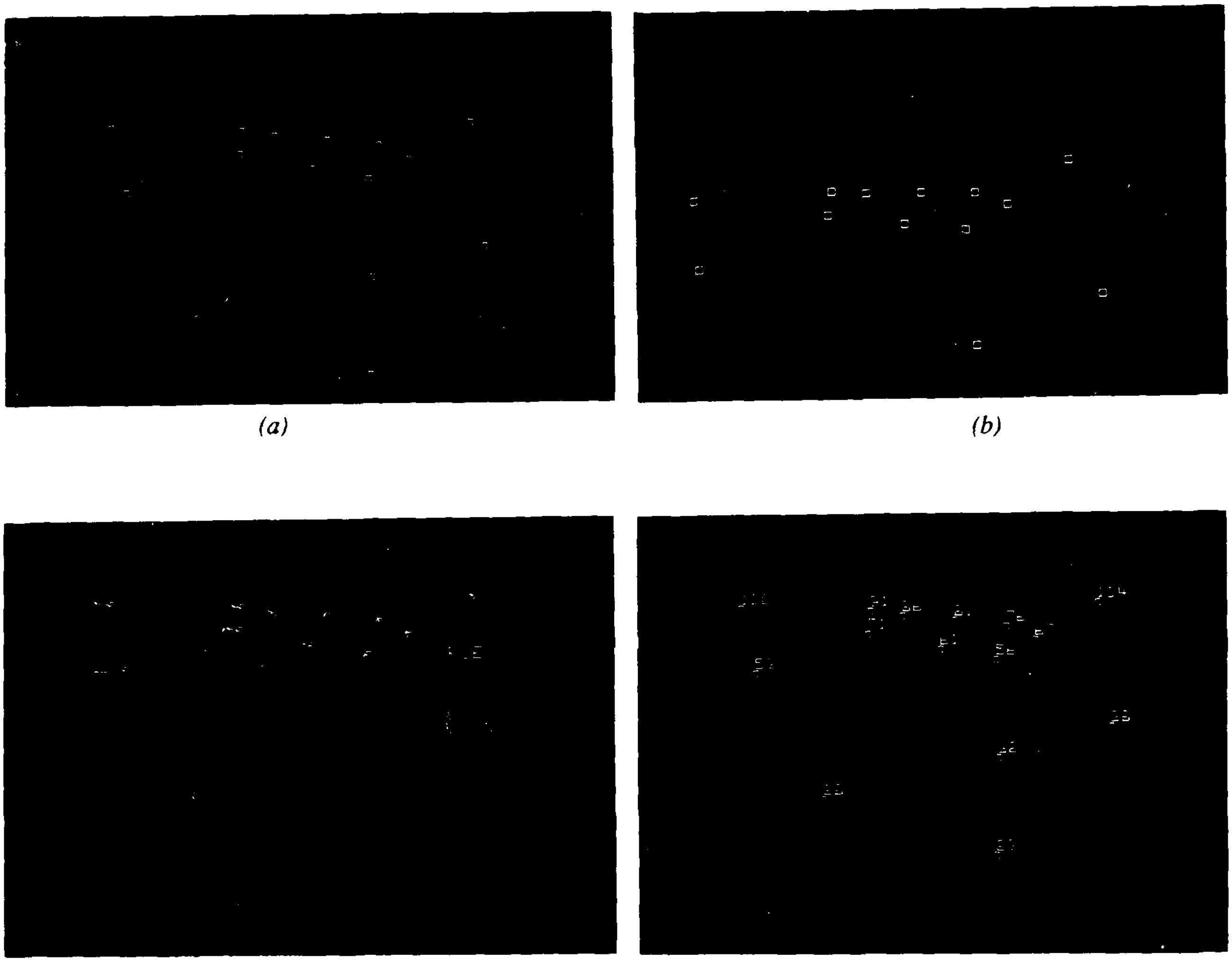
We have presented out initial work for INS integrated motion analysis. Future work will involve incorporating context dependent qualitative scene analysis, knowledge-based sensor management, and incorporation of Kalman filtering into our approach, as shown in Figure 2. Our ultimate goal is to develop the complete, fieldable system for obstacle detection during rotorcraft low altitude flight. We are also applying this technology for land vehicle applications to achieve robust obstacle detection, target motion detection, and target tracking.

	x (ft)	y (ft)	z (ft)
1	100	25	4
2	95	-30	4
3	90	-10	4
4	85	-5	4
5	80	2	4
6	75	8	4
7	70	-8	4
8	65	10	4
9	60	0	4
10	55	5	4
11	50	-15	4
12	35	10	4
13	30	3	4
14	25	-5	4
15	20	2	4

Table 1: Locations of interest points.

	x (ft)	y (ft)	z (ft)	roll (deg)	pitch (deg)	yaw (deg)
Frame A	0	0	-7	0	-15	0
Frame B	5	1	-6	5	-11	2

Table 2: Location, roll, pitch, and yaw of the camera for synthetic frames A and B.



(a) (b)

(c) (d)

Figure 9: Optical flow results using synthetic data. *(a)* Locations of interest points in the first image, indicated by circles. *(b)* Locations of interest points in the second image, shown using squares. Diamonds indicate the derotated interest point locations. *(c)* Matching process results in displacement vectors between circles and diamonds. The FOE is indicated by a cross. *(d)* Computed range values to the interest points.

	x (ft)	y (ft)	z (ft)	roll (deg)	pitch (deg)	yaw (deg)
Frame A	-230.3	-20.72	6.43	0.959	-1.179	-176.737
Frame B	-231.7	-20.83	6.44	1.222	-1.231	-176.852

Table 3: Location, roll, pitch, and yaw of the camera for two frame of real imagery.

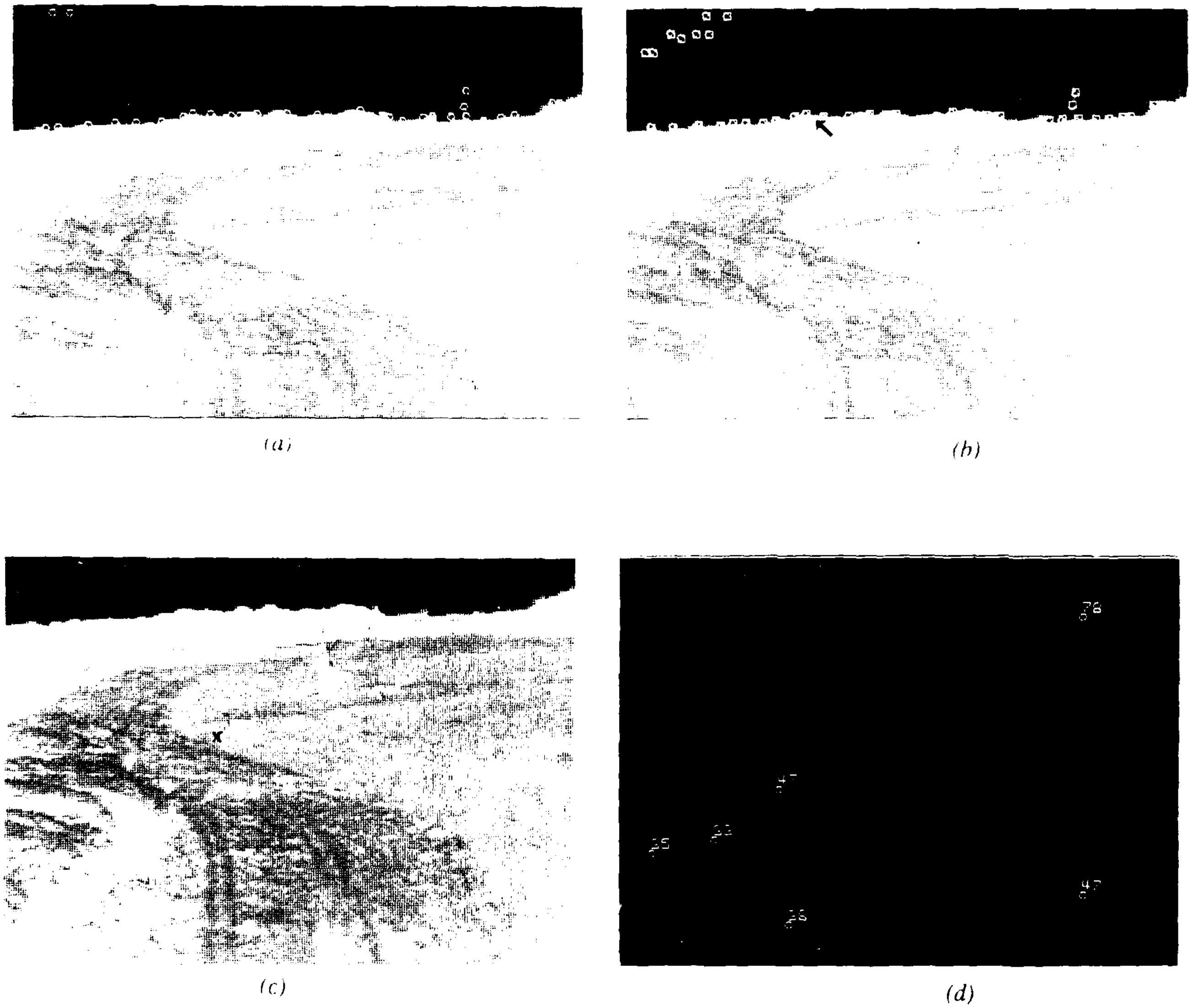


Figure 10: Optical flow results using real data. *(a)* Locations of interest points in the first image, indicated by circles. *(b)* Locations of interest points in the second image, shown using squares. Diamonds indicate the derotated interest point locations. *(c)* Matching process results in displacement vectors between circles and diamonds. The FOE is indicated by a cross. *(d)* Computed range values to the interest points.

ACKNOWLEDGEMENTS

This material is based upon work supported by NASA under contract NAS2-12800 and DARPA/U.S. Army ETL contract DACA 76-86-C-0017. The authors would like to thank Scott Snyder, Dave Duncan, and Paul Gader for discussions and their input to this work.

REFERENCES

1. "Millimeter-Wave Radar May Enhance Safety of Helicopter Flights," *Aviation Week and Space Technology*, p. 103 (July 1987).

2. H. Akimo, "A Method of Bivariate Interpolation and Smooth Surface Fitting for Irregularly Distributed Data Points," *ACM Transactions of Mathematical Software* 4(2) pp. 148-159 (June 1978).

3. H.H. Al-Khatib, "Laser and MMW Backscatter of Transmission Cables," SPIE Proceedings on Physics and Techniques of Coherent Infrared Radar, pp. 212-229 (1981).

4. S.T. Barnard and M.A. Fischler, "Computational Stereo," *ACM Computing Surveys* **14**(4) pp. 553-571 (December 1982).

5. S.T. Barnard and W.B. Thompson, "Disparity Analysis of Images," *IEEE Transactions on Pattern Analysis and Machine Intelligence* **PAMI-2** pp. 333-340 (July 1980).

6. P.J. Besl, "Active Optical Range Imaging Sensor," *Machine Vision and Applications* **1**(2) pp. 127-152 (1988).

7. B. Bhanu and W. Burger, "Approximation of Displacement Fields Using Wavefront Region Growing," *Computer Vision, Graphics, and Image Processing* **41** pp. 306-322 (1988).

8. B. Bhanu and W. Burger, "Qualitative Motion Detection and Tracking of Targets from a Mobile Platform," Proceedings of DARPA Image Understanding Workshop, pp. 289-318 (April 1988).

9. B. Bhanu and B. Roberts, "Obstacle Detection During Rotorcraft Low-Altitude Flight," Annual Technical Report for NASA-Ames (April, 1989).

10. B. Bhanu, P. Symosek, H. Nasr, J. Ming, W. Burger, J. Kim, and H. Lai, "Qualitative Motion Detection and Tracking," Proceedings of DARPA Image Understanding Workshop (May, 1989).

11. C.L. Bowman and M. Gross, "Multi-Sensor Track Association Using Kinematics and Attributes," *Proc. IEEE National Aerospace and Electronics Conference*, pp. 204-208 (1985).

12. B.F. Buxton and H. Buxton, "Low_level Edge Detection for the Extraction of Optical Flow Data and Passive Ranging," *GEC J. Res. Incorporated, Marconi Rev.* **1**(3) pp. 184-187 (1983).

13. V.H.L. Cheng, "Obstacle-Avoidance Automatic Guidance - A Concept-Development Study," Proceedings of AIAA Guidance, Navigation and Control Conference, pp. 1142-1152 (August 1988).

14. M.J. Daily, J.G. Harris, and K. Reiser, "Detecting Obstacles in Range Imagery," Proceedings of DARPA Image Understanding Workshop, pp. 87-97 (February 1987).

15. L. DeFloriani, B. Falcidieno, and C. Pienovi, "Delaunay-based Representation of Surfaces Defined Over Arbitrarily Shaped Domains," *Computer Vision, Graphics, and Image Processing* **32** pp. 127-140 (1985).

16. R. Dutta, R. Manmatha, E.M. Riseman, and M.A. Snyder, "Issues in Extracting Motion Parameters and Depth from Approximate Translational Motion," Proceedings of DARPA Image Understanding Workshop, pp. 945-960 (April, 1988).

17. J.Q. Fang and T.S. Huang, "Some Experiments on Estimating the 3-D Motion Parameters of a Rigid Body from Two Consecutive Image Frames," *IEEE Transactions on Pattern Analysis and Machine Intelligence* **PAMI-6** pp. 545-554 (1984).

18. E.C. Hildreth, "Computations Underlying the Measurement of Visual Motion," *Artificial Intelligence* **23** pp. 309-354 (1984).

19. F.H. Hollister, "Bearings-Only Passive Ranging Using Kalman-Bucy and Moore-Penrose Methods," SPIE Proc. Infrared Technology for Target Detection and Classification, pp. 152-157 (August 1981).

20. B.K.P. Horn and B.G. Schunck, "Determining Optical Flow," *Artificial Intelligence* **17** pp. 185-203 (1981).

21. R.M. Inigo, E.S. McVey, B.J. Berger, and M.J. Wirtz, "Machine Vision Applied to Vehicle Guidance," *IEEE Transactions on Pattern Analysis and Machine Intelligence* **PAMI-6** pp. 820-826 (November 1984).

22. J.K. Kearney and W.B. Thompson, "Gradient-Based Estimation of Optical Flow with Global Optimization," Proc. First conf. Artificial Intell. Applications, pp. 376-380 (December 1984).

23. J.K. Kearney, W.B. Thompson, and D.L. Boley, "Optical Flow Estimation: An Error Analysis of Gradient-Based Methods with Local Optimization," *IEEE Trans. Pattern Analysis and Machine Intelligence* **PAMI-9**(2) pp. 229-244 (March 1987).

24. D.N. Lee, "The Optic Flow Field: The Foundation of Vision," *Phil. Transactions of the Royal Society of London* **B290** pp. 169-179 (1980).

25. H.C. Longuet-Higgins and K. Prazdny, "The Interpretation of a Moving Retinal Image," Proceedings of the Royal Society of London, pp. 385-397 (1980).

26. D. Marr and T. Poggio, "A Computational Theory of Human Stereo Vision," *Proceedings of the Royal Society of London* **B204** pp. 301-328 (1979).

27. L. Matthies, R. Szeliski, and T. Kanade, "Kalman Filter-Based Algorithms for Estimating Depth from Image Sequences," Proceedings of DARPA Image Understanding Workshop, pp. 199-213 (April, 1988).

28. A. Mitiche, S. Seida, and J.K. Aggarwal, "Determining Position and Displacement in Space from Images," Proceedings of IEEE Conference on Computer Vision and Pattern Recognition, pp. 504-509 (June, 1985).

29. K. Prazdny, "Determining the Instantaneous Direction of Motion from Optical Flow Generated by a Curvilinear Moving Observer," *Computer Vision, Graphics, and Image Processing* **17** pp. 238-248 (1981).

30. J.H. Rieger, "Information in Optical Flows Induced by Curved Paths of Observation," *Journal of the Optical Society of America* **73** pp. 339-344 (March 1983).

31. R.E. Sampson, "3-D Range Sensor - Phase Shift Detection," *Special Issue of IEEE Computer on CAD-Based Robot Vision*, pp. 23-24 (August 1987).

32. M.A. Shah and R. Jain, "Detecting Time-Varying Corners," *Computer Vision, Graphics, and Image Processing* **28** pp. 345-355 (1984).

33. H. Shariat and K.E. Price, "How to Use More Than Two Frames to Estimate Motion," Proceedings of IEEE Workshop on Motion, pp. 119-124 (May, 1986).

34. F.W. Smith and M. Streicker, "Passive Ranging from a Moving Vehicle via Optical Flow Measurement," *SPIE: Applications of Digital Image Processing* **829** pp. 310-317 (1987).

35. M.A. Snyder, "The Accuracy of 3-D Parameters in Correspondence-Based Techniques: Startup and Updating," Proc. of the Workshop on Motion, pp. 53-59 (May, 1986).

36. B. Sridhar, V.H.L. Cheng, and A.V. Phatak, "Kalman Filter Based Range Estimation for Autonomous Navigation Using Imaging Sensors," Proceedings of 11th IFAC Symposium on Automatic Control in Aerospace, Tsukuba, Japan (July 1989).

37. C. Thorpe, S. Schafer, and T. Kanade, "Vision and Navigation for the Carnegie Mellon Navlab," Proceedings of DARPA Image Understanding Workshop, pp. 143-153 (February 1987).

APPENDIX A

NAVIGATION ERROR CHARACTERISTICS

The passive ranging technique relies on attitude and position information provided by the inertial navigation system (INS). The purpose of this appendix is to supply error models which are representative of a land vehicle/helicopter navigation system. These error models can then be used to assess the impact of navigational errors on the passive ranging performance. A short term navigation error model of a strapdown inertial navigation system has been developed. The INS derives attitude, velocity, and position based on inputs from orthogonal triads of gyros and accelerometers. Assuming that the only error of concern to the passive ranging technique is the drift between samples, Table A1 provides error estimates for the GG1328 Ring Laser Gyro (RLG) INS and the GG1342 RLG INS systems. The error estimates in the table are based on the following assumptions:

30 Hz Sample Rate
45 Degree Latitude and Heading
10 Minute Gyrocompass Alignment
0.5 g X and Y Helicopter Acceleration
1.0 g Z Helicopter Acceleration
100 deg/sec Roll, Pitch, and Yaw Rates
H-764 Output Quantization Levels

As evident from Table A1, the most significant one sigma error source is the output quantization error. The output quantization is a function of output requirements as internally the attitudes, velocities, and positions are all double precision quantities. Thus, the small errors in attitude and velocity vectors obtained from the INS allow the location of the focus of expansion (FOE) in the original and derotated images to be accurate enough to perform robust motion analysis. Further details of this analysis are given in the report by Bhanu and Roberts.[8]

	Level Error (mrad)		Heading Error (mrad)	
Error Term	GG1328	GG1342	GG1328	GG1342
Gyro Bias	0.1600e-04	0.1600e-05	0.1600e-04	0.1600e-05
Gyro Random Walk	0.1585e-02	0.2642e-03	0.1585e-02	0.2642e-03
Gyro Scalefactor	0.1728e-02	0.8639e-03	0.1728e-02	0.8639e-03
Gyro Misalignment	0.9647e-04	0.6431e-04	0.9647e-04	0.6431e-04
Gyro Misalignment	0.9647e-04	0.6431e-04	0.9647e-04	0.6431e-04
Heading Error	-.1404e-07	-.1790e-08		
RSS Total	0.2349e-02	0.9080e-03	0.2345e-02	0.9034e-03
Output Quantization	0.0141e-00	0.0141e-00	0.0141e-00	0.0141e-00
RSS Total	0.0143e-00	0.0141e-00	0.0143e-00	0.0141e-00

	X - Y Velocity Error (fps)		Z Velocity Error (fps)	
Error Term	GG1328	GG1342	GG1328	GG1342
Accel Bias	0.1063e-03	0.5313e-04	0.1063e-03	0.5313e-04
Accel Scalefactor	0.1063e-03	0.5313e-04	0.2125e-03	0.1063e-03
Accel Misalignment	0.2576e-04	0.1288e-04	0.2576e-04	0.1288e-04
Accel Misalignment	0.1030e-03	0.5152e-04	0.2576e-04	0.1288e-04
Level Error	0.1063e-03	0.5314e-04		
RSS Total	0.2125e-03	0.1063e-03	0.2404e-03	0.1202e-03
Output Quantization	0.1409e-03	0.1409e-03	0.1409e-03	0.1409e-03
RSS Total	0.2550e-03	0.1765e-03	0.2786e-03	0.1852e-03

	X - Y Position Error (feet)		Z Position Error (feet)	
Error Term	GG1328	GG1342	GG1328	GG1342
Accel Bias	0.3507e-05	0.1753e-05	0.3507e-05	0.1753e-05
Accel Scalefactor	0.3507e-05	0.1753e-05	0.7013e-05	0.3507e-05
Accel Misalignment	0.8500e-06	0.4250e-06	0.8500e-06	0.4250e-06
Accel Misalignment	0.3400e-05	0.1700e-05	0.8500e-06	0.4250e-06
Level Error	0.3508e-05	0.1754e-05		
RSS Total	0.7013e-05	0.3506e-05	0.7933e-05	0.3966e-05
Output Quantization	0.2528e-00	0.2528e-00	0.2528e-00	0.2886e-00
RSS Total	0.2528e-00	0.2528e-00	0.2528e-00	0.2886e-00

Table A1: One Sigma, Short Term Navigation Errors

Robust Geometric Computations for Vision and Robotics *

Victor J. Milenkovic
Computer Science Department
Carnegie Mellon University
Pittsburgh, PA 15213

Abstract

A new technique, the *hidden variable method*, is described for the creation of *robust* geometric algorithms: algorithms which can be implemented with absolute reliability using rounded finite precision arithmetic. The specific application given here is an algorithm for computing arrangements of line segments in two dimensions: determining how a set of line segments subdivides the plane into vertices, line segments, and polygons. Using *rubber band curves*, an implicit representation for the line segments, the algorithm can generate an arrangement with N bits of accuracy using rounded arithmetic with just $N + 6$ bits of precision. Applications of this algorithm to problems in vision and robotics are discussed.

1 Introduction

Shape, position, orientation, and velocity are all geometric properties, and reasoning about these properties is an important part of vision and robotics. As much as possible, we wish to automate geometric reasoning by means of geometric programs. In order that these programs be efficient, we use rounded finite precision arithmetic. Unfortunately, when we write these programs we find that it is difficult to attain reasonable reliability and almost impossible to obtain absolute reliability: there always seems to be one more special case on which the program fails. Theoretical reasons are only now coming to light as to why geometric programs are so much more difficult to make reliable than purely numerical or purely symbolic programs, but this difficulty is very commonly experienced in practice.

We propose techniques for creating *robust* geometric programs: geometric programs with absolutely reliable rounded arithmetic implementations. This paper will focus on the domain of line segments in the plane, but the techniques presented here have broader applications. The techniques are based on two principles. First, one should use a malleable representation. Specifically, replace each line segment with a *rubber band curve* which can be modified as the computation proceeds to reconcile numerical error with the symbolic structure. Second, one should keep as much of the representation implicit as is possible. In the case of rubber band curves, their exact shapes are unknown at all stages of the computation. Put together, these two principles make up the *hidden variable method* which allows us to generate correct geometric information without the use of exact arithmetic. A certain amount of error is introduced by the use of rubber band curves, but this can be made to be a small fraction of the error arising from sensor noise and other measurement errors.

1.1 Outline

Section 2 reviews the pitfalls of using rounded arithmetic to do geometry. The cost of using exact arithmetic instead is also considered. Section 3 formalizes the concept of single-precision arithmetic and explains why we expect robust single-precision geometric algorithms to exist. Section 4 gives a robust single-precision algorithm for reasoning about line segments in the plane. Given a set of line segments, we can determine

*This research was supported in part by DARPA, monitored by the Air Force Avionics Lab under contract F33615-87-C-1499. Victor J. Milenkovic was supported in part by an IBM Manufacturing Research Fellowship. The views and conclusions contained in this document are those of the authors and should not be interpreted as representing the official policies, either expressed or implied, of the funding agencies. Author's current address: Aiken Computation Lab, Harvard University, Cambridge MA 02138.

how they intersect and what polygons they form using rounded arithmetic. This algorithm avoids the pitfalls discussed in Section 2, and it can be used for a variety of applications, such as modeling polygonal regions in the plane.

1.2 Background

In order to qualify as *robust* under the definitions in the paper, a geometric algorithm must be implementable using rounded $(N+c)$-bit arithmetic where N is the number of bits of accuracy in the output and c is constant or perhaps logarithmic in N. The implementation is not allowed to simulate high precision arithmetic, and it generates a solution for every well-formed input. The author has also created line arrangement and plane arrangement algorithms that satisfy this definition [10].

For a discussion of the more general problem of improving the reliability geometric computations using rounded arithmetic, see [4,5]. Problems addressed include polygon [13,9] and polyhedron [6,7,14] modeling and the calculation of convex hulls [12] and other problems in plane geometry [11]. Karasick [7] has the most practical result to date, an ultra-reliable polyhedral modeling system that is provably *safe*: either it generates a topologically consistent result or it indicates failure. Greene and Yao [3] have addressed the problem of rounding the output of an exact algorithms, and rounding can also be accomplished using the techniques in [10] and [11].

2 Pitfalls of Using Rounded Arithmetic

The problems that arise as a result of using rounded arithmetic to implement geometric algorithms fall into four areas: incomplete representation, ill-conditioned problems, incomplete information, and unexpected incidences. Problems arising from these areas cause either unreliable behavior and/or unbounded error. This section will consider each one in turn.

2.1 Incomplete Representation

Suppose we have a pair of line segments **AB** and **CD** whose endpoints are represented using standard single precision floating point numbers. In general, the intersection **I** of **AB** and **CD** will not be representable in single precision. It will in fact be a fraction, and if we carry out the division, the result is a repeating decimal.

One way out of this difficulty is to round **I** to the nearest representable point. Thus we replace **AB** by **AI** and **IB** and similarly for **CD**. This replacement can have two effects. First, it adds error which is ultimately proportional to the number of segments that **AB** intersects. Having error grow linearly with the problem size is usually not practical.

Rounding **I** may also introduce new intersections. In particular, **AI** and **IB** may intersect a segment **EF** which was not intersected by **AB**. Actually, the algorithm of Section 4 has this property. The point is that we must be careful to detect the consequences of the changes we make.

2.2 Ill-Conditioned Problems

If **AB** and **CD** intersect at a very small angle, it requires at least triple precision arithmetic (three times input precision) in order to calculate **I** to the same accuracy as **A**, **B**, **C** and **D**. As Section 3 will show, we should not need so much precision to generate a reasonable solution to the problem. However, from a naive point of view, a solution in which **I** is calculated with less accuracy than **A** is incorrect.

Small angle intersections are common in practice because **AB** and **CD** may be equal or overlapping segments which have been perturbed by measurement error or rounding.

2.3 Incomplete Information

Three segments $\mathbf{A}_1\mathbf{B}_1$, $\mathbf{A}_2\mathbf{B}_2$, and $\mathbf{A}_3\mathbf{B}_3$ can intersect in essentially three different ways: $\mathbf{I}_{12}$ (the intersection of $\mathbf{A}_1\mathbf{B}_1$ and $\mathbf{A}_2\mathbf{B}_2$) lies left of, right of, or on $\mathbf{A}_3\mathbf{B}_3$. In order to correctly determine which case holds, we must use more than six times the input precision for our calculations. If we do not use this much precision,

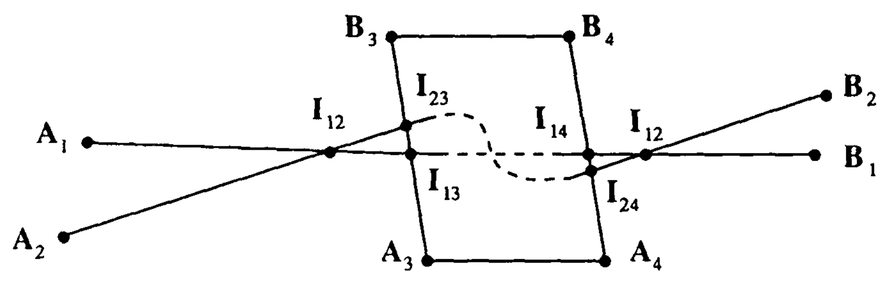


Figure 1: Global Contradiction

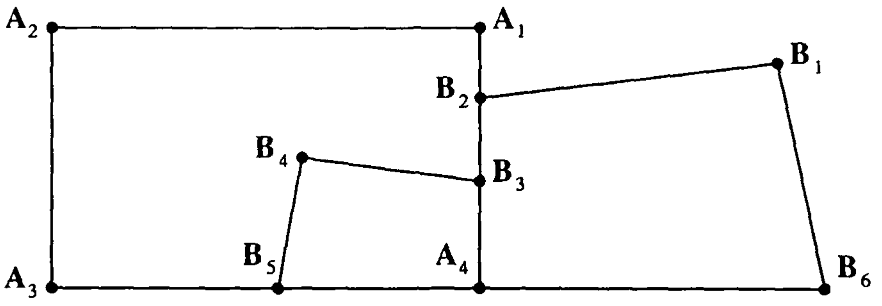


Figure 2: Unexpected Incidence of $\mathbf{A}_3\mathbf{A}_4$ and $\mathbf{B}_5\mathbf{B}_6$

we are left with situations in which we cannot know the correct answer, and thus one consequence of rounding is incomplete information.

Let us see what can happen when we make an arbitrary choice in the case that we do not know the correct answer. Figure 1 shows what happens when two *local* choices, $\mathbf{I}_{12}$ lies to the left of $\mathbf{A}_3\mathbf{B}_3$ and to the right of $\mathbf{A}_4\mathbf{B}_4$, lead to a global contradiction. If we attempt to trace out the boundary of the quadrilateral $\mathbf{A}_3\mathbf{A}_4\mathbf{I}_{24}\mathbf{I}_{23}$, we end up with the following sequence: $\mathbf{A}_3$, $\mathbf{A}_4$, $\mathbf{I}_{24}$, $\mathbf{I}_{23}$, $\mathbf{I}_{13}$, $\mathbf{I}_{14}$ (!), $\mathbf{B}_4$, $\mathbf{B}_3$, $\mathbf{I}_{23}$, $\mathbf{I}_{24}$, $\mathbf{I}_{14}$, $\mathbf{I}_{13}$, and (finally) back to $\mathbf{A}_3$. This is clearly an erroneous solution resulting from that fact that the local choices we have made have caused an unacknowledged intersection between segments $\mathbf{I}_{23}\mathbf{I}_{24}$ and $\mathbf{I}_{13}\mathbf{I}_{14}$. If we are to avoid this error, we must either avoid causing extra intersections or we must detect them.

2.4 Unexpected Incidences

In Figure 2 we see two polygons $\mathbf{A}_1\mathbf{A}_2\mathbf{A}_3\mathbf{A}_4$ and $\mathbf{B}_1\mathbf{B}_2\mathbf{B}_3\mathbf{B}_4\mathbf{B}_5\mathbf{B}_6$. Suppose we attempt to take the union using a *covered edge* rule: remove any edge covered by the other polygon. Edge $\mathbf{A}_4\mathbf{B}_5$ is duplicated, and if we naively apply the rule, either we remove both copies or we remove neither. Neither solution is satisfactory.

This problem arises because the rule does not cover a special case: overlapping edges. This failure, which we refer to as an *unexpected incidence*, can occur whether we are using exact or rounded arithmetic. For the domain of exact arithmetic, there is a general technique called *simulation of simplicity* [2,15] for removing these special cases. In the domain of rounded arithmetic, we have no choice but to use careful programming.

3 Accuracy and Precision

In this section we will discuss the issues of precision and accuracy. We define single-precision arithmetic and show that it is sufficient to generate accurate solutions to numerical problems. We will also see why it is hard to reliably compute single-precision solutions to geometric problems.

3.1 Definitions

Definition 1 (Accuracy) *Suppose $a \in \Re$ is an approximate input to a numerical computation. If $a_{correct}$ is the correct value of a, we say that a has N bits of* accuracy *if,*

$$|a - a_{correct}| \leq 2^N a.$$

A number $\xi \in \Re$ is small *with respect to a if $|\xi| \leq 2^N a$.*

Definition 2 (P-bit Arithmetic) *An arithmetic has P bits of* precision *if any single operation has P bits of accuracy. For example,*

$$|(a+b)_{\mathrm{R}} - (a+b)| \leq 2^P |a+b|,$$

where the subscript R denotes the rounded operation.

For a standard floating point arithmetic, P is the number of bits in the mantissa. We will say that a computation uses *single-precision* arithmetic if it solves problems with N bits of accuracy using P-bit arithmetic, where $P \approx N$.

3.2 Non-Geometric Numerical Computation

As an example of a non-geometric numerical computation, consider the problem of solving a system of D linear equations in D variables. Using Gaussian elimination and single-precision arithmetic, we can safely and efficiently generate a reasonable solution.

What do we mean by "safe"? The order in which Gaussian elimination eliminates variables depends on what appears to be the largest coefficient in a column. Round-off error may increase the value of a coefficient and thus alter the order in which variables are eliminated. In other words, Gaussian elimination acts on the basis of incomplete information. Fortunately, no matter how the order comes out, the algorithm generates a reasonable result.

What is efficient? If the coefficients have N bits of accuracy, arithmetic with $P = N + 2\log D$ bits suffices. Since $\log D$ is small, this is basically "single" precision, $P \approx N$. In order to solve the system exactly, we would require at least $P = DN$ bits, D-tuple precision. Furthermore, the rounded solution uses no more operations than the exact solution, and obviously, single-precision operations are much cheaper than D-tuple precision operations. We cannot hope to generate a solution with N bits of accuracy using arithmetic with $P < N$, and therefore the rounded solution is as efficient as possible.

What is reasonable? In the case of an ill-conditioned problem, the single-precision implementation of Gaussian elimination may generate a solution point that is very far from the correct solution. However, one can prove that if the approximate single-precision solution is $\mathbf{x} = \langle c_1, c_2, \ldots, c_D \rangle$, then there exists some small (Definition 1) perturbation of the coefficients such that $\mathbf{x}$ is the exact infinite-precision solution to the perturbed system. Thus, given the accuracy to which we know the input coefficients, $\mathbf{x}$ may in fact be the correct solution.

3.3 Geometric Problems

Our goal is to safely and efficiently generate reasonable solutions to geometric problems using single-precision arithmetic. Arbitrary decisions in the face of incomplete information do not seem to have a significant effect on Gaussian elimination. Unfortunately, arbitrary decisions do have a strong effect on geometric algorithms, as we have seen in Section 2. Because of topological constraints, the decisions are highly interdependent.

Because geometric reasoning is so sensitive to arbitrary decisions, we must be careful how we define a reasonable solution. In the case of line segments, one might think that we should allow each endpoint to vary within a small disk, but this still appears to be too constrained. Recent theoretical work in computational geometry [1] has shown that determining whether a particular topology can be generated by a set of segments is a very hard problem.

Instead, we will fix the endpoints and allow the interior of the segment to vary into the shape of an arbitrary monotonic curve. As we will see in the next section, we can still constrain the shape of the approximation to be roughly linear. This type of solution is reasonable because we can constrain the points of the curve to approximate the points of the segment to N bits of accuracy. The solution is physically meaningful because straight lines do not occur in the natural world. A sufficiently straight curve is as useful a model as a straight line segment for most purposes.

4 Robust Reasoning about Line Segments

This section describes a technique for reasoning about line segments using an implicit approximation called the *rubber band curve.* After a few basic definitions, we will describe a straightforward means for storing topological information called a SIDE database. By imposing certain consistency conditions on the arbitrary decisions we store in the database we can assure that there exist a set of rubber band curves consistent with the decisions we have made. Next, we see how to detect implicit intersections among these curves by examining the values in the database. This section ends with an algorithm that creates new vertices and moves the intersection points of the rubber band curves to these explicit vertex locations.

4.1 Definitions

Definition 3 *A* coordinate rectangle *is a rectangle whose sides are aligned with the coordinate axes. Let* $\mathbf{AB}$ *be a line segment in the plane (with floating point endpoints). Let* $\mathrm{R}(\mathbf{AB})$ *be the coordinate rectangle which has* $\mathbf{AB}$ *as one of its diagonals. In interval notation,*

$$\mathrm{R}(\mathbf{AB}) = [\mathbf{A}_x, \mathbf{B}_x] \times [\mathbf{A}_y, \mathbf{B}_y].$$

If a vertex $\mathbf{C}$ lies outside $\mathrm{R}(\mathbf{AB})$, then we can easily classify $\mathbf{C}$ with respect to $\mathbf{AB}$. For example if $\mathbf{C}_x \in [\mathbf{A}_x, \mathbf{B}_x]$ and $\mathbf{C}_y > \max(\mathbf{A}_y, \mathbf{B}_y)$, then $\mathbf{C}$ is clearly above $\mathbf{AB}$.

If $\mathbf{C} \in \mathrm{R}(\mathbf{AB})$, we must look at the signed distance $\delta(\mathbf{C}, \mathbf{AB})$ from $\mathbf{C}$ to $\mathbf{AB}$,

$$\delta(\mathbf{C}, \mathbf{AB}) = \frac{(\mathbf{B} - \mathbf{A}) \times (\mathbf{C} - \mathbf{A})}{|\mathbf{B} - \mathbf{A}|},$$

where,

$$(\mathbf{B} - \mathbf{A}) \times (\mathbf{C} - \mathbf{A}) = (\mathbf{B}_x - \mathbf{A}_x)(\mathbf{C}_y - \mathbf{A}_y) - (\mathbf{B}_y - \mathbf{A}_y)(\mathbf{C}_x - \mathbf{A}_x).$$

Suppose we use arithmetic with P bits of precision. How accurate is $\delta(\mathbf{C}, \mathbf{AB})_{\mathrm{R}}$? (The subscript R refers to the use of rounded arithmetic.) It can be shown that for $\mathbf{C}$ close to $\mathbf{AB}$,

$$|\delta(\mathbf{C}, \mathbf{AB})_{\mathrm{R}} - \delta(\mathbf{C}, \mathbf{AB})| \leq 3\epsilon|\mathbf{C} - \mathbf{A}|,$$

where $\epsilon = 2^{-P}$. We will assume that all reasoning occurs inside some bounding square $[-\mathrm{M}, \mathrm{M}] \times [-\mathrm{M}, \mathrm{M}]$ where M is the maximum magnitude of any coordinate. Thus $|\mathbf{C} - \mathbf{A}| \leq 2\sqrt{2}\mathrm{M}$, and the error in calculating $\delta(\mathbf{C}, \mathbf{AB})$ is bounded by $6\sqrt{2}\epsilon\mathrm{M}$. We will refer to this constant as α.

4.2 Handling Incomplete Information

The fact that $\alpha \neq 0$ indicates that for $|\delta(\mathbf{C}, \mathbf{AB})_{\mathrm{R}}| \leq \alpha$, we do not know whether $\mathbf{C}$ lies above, below or on $\mathbf{AB}$. We make arbitrary decisions and store them in a SIDE database: for each segment $\mathbf{AB}$ and each vertex $\mathbf{C} \in \mathrm{R}(\mathbf{AB})$, $\mathrm{SIDE}(\mathbf{C}, \mathbf{AB})$ equals ABOVE, BELOW or ON. Since $\mathbf{A}$, $\mathbf{B}$, $\mathbf{C}$ are vertices we are given or ones we create, the database stores only a finite amount of information. The decisions we make may be incorrect, but by imposing some consistency conditions, we can still generate useful results.

Definition 4 (Numerical Consistency) *The SIDE database is* numerically consistent *if*

$$\mathrm{SIDE}(\mathbf{P}, \mathbf{AB}) = \left\{ \begin{array}{c} \mathrm{ABOVE} \\ \mathrm{BELOW} \\ \mathrm{ON} \end{array} \right\} \textit{ implies that } \left\{ \begin{array}{c} \delta(\mathbf{C}, \mathbf{AB}) \geq -5\alpha \\ \delta(\mathbf{C}, \mathbf{AB}) \leq 5\alpha \\ |\delta(\mathbf{C}, \mathbf{AB})| \leq 5\alpha \end{array} \right\}.$$

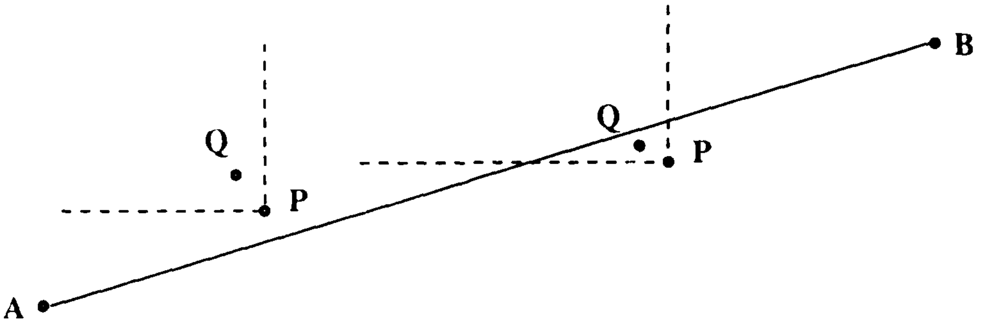


Figure 3: Examples of XY-Consistency

This definition states that the decisions may be incorrect, but not very incorrect. The reason we use a multiple 5α will become apparent later. The value 5α represents about six bits of error.

As we know from Section 2, imposing arbitrary decisions in the face of incomplete information is a risky business. Therefore we enforce another consistency condition on SIDE.

Definition 5 (XY-Consistency: Prototype Definition) *Suppose we have* $\mathbf{A}_x < \mathbf{B}_x$ *and* $\mathbf{A}_y < \mathbf{B}_y$ *and* $\mathbf{P}, \mathbf{Q} \in \mathrm{R}(\mathrm{AB})$. *If* $\mathrm{SIDE}(\mathbf{P}, \mathbf{AB}) = \mathrm{ABOVE}$ *and if* $\mathbf{Q}_x \le \mathbf{P}_x$ *and* $\mathbf{Q}_y \ge \mathbf{P}_y$, *then* $\mathrm{SIDE}(\mathbf{Q}, \mathbf{AB}) = \mathrm{ABOVE}$ *(see Figure 3).*

Imagine putting the origin at **P**. The fact that **P** is ABOVE **AB** forces everything in the second quadrant to be ABOVE also. Even if we are wrong about $\mathrm{SIDE}(\mathbf{P}, \mathbf{AB})$, we will still insist on xy-consistency. The following is the complete definition.

Definition 6 (XY-Consistency: Complete Definition) *Suppose we have* $\mathbf{A}_x < \mathbf{B}_x$ *and* $\mathbf{A}_y < \mathbf{B}_y$ *and* $\mathbf{P}, \mathbf{Q} \in \mathrm{R}(\mathbf{AB})$.

$$\textit{If}\ \mathrm{SIDE}(\mathbf{P}, \mathbf{AB}) = \left\{ \begin{array}{c} \mathrm{ON}\ \textit{or}\ \mathrm{ABOVE} \\ \mathrm{ON}\ \textit{or}\ \mathrm{BELOW} \end{array} \right\}\ \textit{and} \left\{ \begin{array}{c} \mathbf{Q}_x \le \mathbf{P}_x \\ \mathbf{Q}_x \ge \mathbf{P}_x \end{array} \right\}\ \textit{and} \left\{ \begin{array}{c} \mathbf{Q}_y \ge \mathbf{P}_y \\ \mathbf{Q}_y \le \mathbf{P}_y \end{array} \right\}$$

$$\textit{then}\ \mathrm{SIDE}(\mathbf{Q}, \mathbf{AB}) = \left\{ \begin{array}{c} \mathrm{ON}\ \textit{or}\ \mathrm{ABOVE} \\ \mathrm{ON}\ \textit{or}\ \mathrm{BELOW} \end{array} \right\}.$$

If either of these two conditions holds, $\mathrm{SIDE}(\mathbf{Q}, \mathbf{AB})$ *can equal* ON *only in the case that*

$$\mathrm{SIDE}(\mathbf{P}, \mathbf{AB}) = \mathrm{ON} \qquad \textit{and} \qquad (\mathbf{Q}_x = \mathbf{P}_x\ \textit{or}\ \mathbf{Q}_y = \mathbf{P}_y).$$

Finally, if $\mathbf{A}_y > \mathbf{B}_y$ *the definition is the same except that comparisons of y-coordinates are reversed.*

4.3 Rubber Band Curves

The following theorem indicates that there is some reasonable curve satisfying the set of decisions we have made.

Theorem 1 (Hidden Variable Theorem) *Let* $\gamma : [0,1] \to \Re^2$ *be the shortest rectifiable (actually polygonal) curve satisfying* $\gamma(0) = \mathbf{A}$, $\gamma(1) = \mathbf{B}$, *and for all vertices* $\mathbf{P} \in \mathrm{R}(\mathbf{AB})$,

$$\mathrm{SIDE}(\mathbf{P}, \mathbf{AB}) = \left\{ \begin{array}{c} \mathrm{ABOVE} \\ \mathrm{BELOW} \\ \mathrm{ON} \end{array} \right\}\ \textit{implies that}\ \mathbf{P}\ \textit{is} \left\{ \begin{array}{c} \textit{above or on} \\ \textit{below or on} \\ \textit{on} \end{array} \right\} \gamma.$$

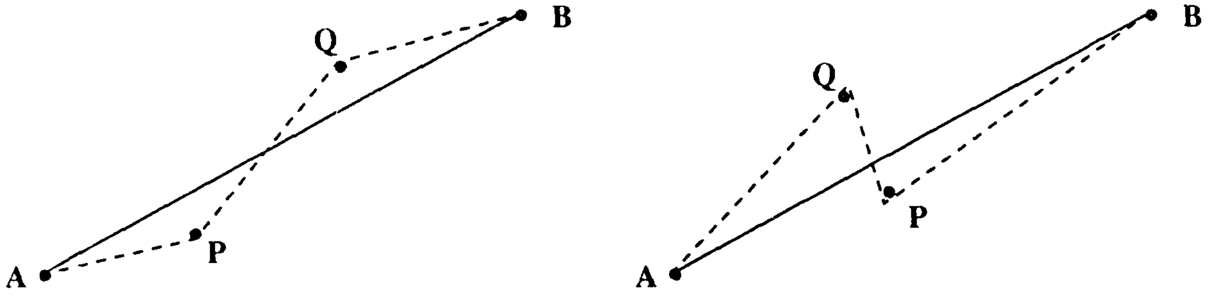


Figure 4: Rubber Band Curves

Then γ is monotonic and close to $\mathbf{AB}$:

$$\forall t \in [0,1]: \quad \gamma'(t)\cdot(\mathbf{B}-\mathbf{A}) > 0 \quad \textit{and} \quad |\delta(\gamma(t),\mathbf{AB})| \leq 5\alpha.$$

The tangent $\gamma'(t)$ may be undefined at a finite number of points.

For a proof, see [10]. We call the curve γ the *rubber band curve* rbc($\mathbf{AB}$). Figure 4 gives two examples in which $\mathbf{P}$ is ABOVE $\mathbf{AB}$ and $\mathbf{Q}$ is BELOW. The instance on the right shows what happens when xy-monotonicity is violated; the rubber band curve (dashed) is not monotonic.

4.4 Detecting Intersections

Rubber band curves are implicit. In the examples in Figure 4 we cannot know if $\mathbf{P}$ and $\mathbf{Q}$ actually deflect rbc($\mathbf{AB}$) unless we know the signs of $\delta(\mathbf{P},\mathbf{AB})$ and $\delta(\mathbf{Q},\mathbf{AB})$, which we do not in the case of incomplete information. Despite this implicitness, we can always tell whether or not two rubber band curves intersect simply by looking at the SIDE database. This section shows how to accomplish this.

Definition 7 (Subsegment) $\mathbf{P}_i\mathbf{P}_{i+1}$ *is a (minimal)* subsegment *of segment* $\mathbf{AB}$ *if*

- SIDE($\mathbf{P}_i$, $\mathbf{AB}$) = SIDE($\mathbf{P}_{i+1}$, $\mathbf{AB}$) = ON;
- $\mathbf{P}_i$ *and* $\mathbf{P}_{i+1}$ *are* neighbors: *for every other vertex* $\mathbf{P} \in \mathrm{R}(\mathbf{P}_i\mathbf{P}_{i+1})$, SIDE($\mathbf{P}$, $\mathbf{AB}$) $\neq$ ON.

Define also $\mathrm{rbc}_{\mathbf{AB}}(\mathbf{P}_i\mathbf{P}_{i+1})$ *to be the portion of* rbc($\mathbf{AB}$) *joining* $\mathbf{P}_i$ *to* $\mathbf{P}_{i+1}$.

If a subsegment of $\mathbf{AB}$ crosses a subsegment of $\mathbf{CD}$ we expect to find evidence of the fact in the SIDE database.

Definition 8 (Evidence Vertices) *Let* $\mathbf{P}_i\mathbf{P}_{i+1}$ *be a* subsegment *of segment* $\mathbf{AB}$ *and let* $\mathbf{Q}_j\mathbf{Q}_{j+1}$ *be a* subsegment *of segment* $\mathbf{CD}$ *such that* $\mathrm{R}(\mathbf{P}_i\mathbf{P}_{i+1}) \cap \mathrm{R}(\mathbf{Q}_j\mathbf{Q}_{j+1})$ *is non-empty. Let* $\mathbf{E}$ *be a vertex such that* $\mathbf{E}_x \in [\mathbf{P}_{i,x}, \mathbf{P}_{i+1,x}] \cap [\mathbf{Q}_{j,x}, \mathbf{Q}_{j+1,x}]$.

Vertex $\mathbf{E}$ *is an* evidence vertex *if* SIDE($\mathbf{E}$, $\mathbf{AB}$) $\neq$ SIDE($\mathbf{E}$, $\mathbf{CD}$). *In particular* $\mathbf{E}$ *is* evidence *that* $\mathrm{rbc}_{\mathbf{AB}}(\mathbf{P}_i\mathbf{P}_{i+1})$ *is partially above* $\mathrm{rbc}_{\mathbf{CD}}(\mathbf{Q}_j\mathbf{Q}_{j+1})$ *if*

$$\mathrm{SIDE}(\mathbf{E},\mathbf{AB}) = \mathrm{BELOW} \qquad \textit{or} \qquad \mathrm{SIDE}(\mathbf{E},\mathbf{CD}) = \mathrm{ABOVE}.$$

Otherwise it is evidence *that* $\mathrm{rbc}_{\mathbf{AB}}(\mathbf{P}_i\mathbf{P}_{i+1})$ *is partially below* $\mathrm{rbc}_{\mathbf{CD}}(\mathbf{Q}_j\mathbf{Q}_{j+1})$.

Lemma 1 *Let* $\mathbf{P}_i\mathbf{P}_{i+1}$ *and* $\mathbf{Q}_j\mathbf{Q}_{j+1}$ *be subsegments of* $\mathbf{AB}$ *and* $\mathbf{CD}$*, respectively. Curve* $\mathrm{rbc}_{\mathbf{AB}}(\mathbf{P}_i\mathbf{P}_{i+1})$ *crosses curve* $\mathrm{rbc}_{\mathbf{CD}}(\mathbf{Q}_j\mathbf{Q}_{j+1})$ *if and only if there is an evidence vertex* $\mathbf{E}_a$ *that indicates* $\mathrm{rbc}_{\mathbf{AB}}(\mathbf{P}_i\mathbf{P}_{i+1})$ *is partially above* $\mathrm{rbc}_{\mathbf{CD}}(\mathbf{Q}_j\mathbf{Q}_{j+1})$ *and there is an evidence vertex* $\mathbf{E}_b$ *that indicates* $\mathrm{rbc}_{\mathbf{AB}}(\mathbf{P}_i\mathbf{P}_{i+1})$ *is below* $\mathrm{rbc}_{\mathbf{CD}}(\mathbf{Q}_j\mathbf{Q}_{j+1})$ *(see Figure 5).*

Thus if two subsegments cross, we can detect the fact looking only at the values in the SIDE database. In the left half of Figure 5 $\mathbf{E}_a$ equals $\mathbf{U}_3$ and $\mathbf{E}_b$ equals $\mathbf{U}_1$. This always happens if the slopes of $\mathbf{AB}$ and $\mathbf{CD}$ have the same sign. For a proof of the lemma, see [10].

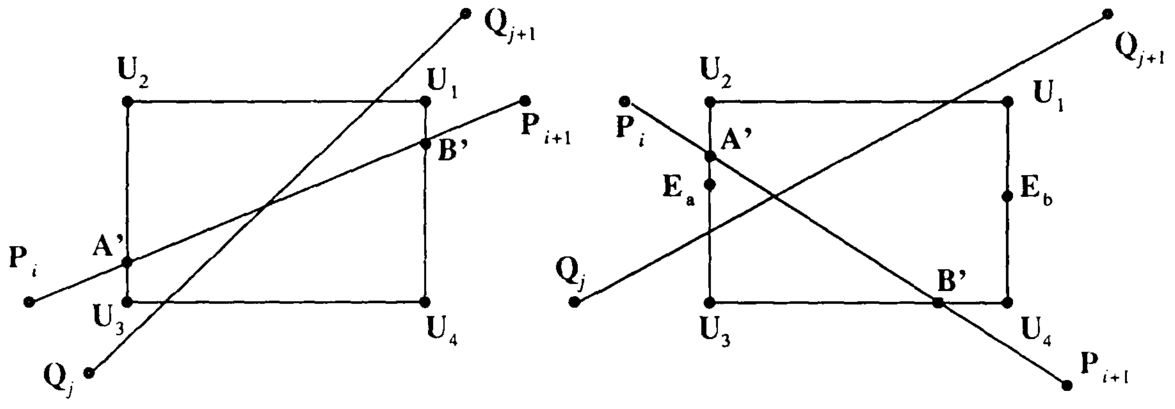


Figure 5: Evidence for Intersection of Subsegments $\mathbf{P}_i\mathbf{P}_{i+1}$ and $\mathbf{Q}_j\mathbf{Q}_{j+1}$

4.5 Creating Intersections

When two subsegments cross we must create an explicit vertex for this crossing and then add it to the SIDE database. Eventually, we will find all intersections, and as the lemma of the previous section indicates, by scrutinizing the SIDE database, we can be sure that we have not missed any intersections. When we have finished, the SIDE database will contain all the topological information we need to know about the rubber band curves. This section describes the intersection algorithm.

Lemma 1 implies the existence of evidence vertices $\mathbf{E}_a$ and $\mathbf{E}_b$ whenever two subsegments $\mathbf{P}_i\mathbf{P}_{i+1}$ and $\mathbf{Q}_j\mathbf{Q}_{j+1}$ intersect. The specific task of the intersection algorithm is to create a, new vertex $\mathbf{I}$ such that $\mathbf{E}_{a,x} \leq \mathbf{I}_x \leq \mathbf{E}_{b,x}$ and such that it is numerically consistent and xy-consistent (Definitions 4 and 6) to set *SIDE*$(\mathbf{I}, \mathbf{AB})$ and *SIDE*$(\mathbf{I}, \mathbf{CD})$ *equal to ON. In order to be xy-consistent*, it is necessary that,

$$\mathbf{I} \in \mathbf{R} = ([\mathbf{E}_{a,x}, \mathbf{E}_{b,x}] \times [-\infty, \infty]) \cap \mathrm{R}(\mathbf{P}_i\mathbf{P}_{i+1}) \cap \mathrm{R}(\mathbf{Q}_j\mathbf{Q}_{j+1}).$$

We call $\mathbf{R}$ the *bounding rectangle for the intersection* and denote its vertices by $\mathbf{U}_1$, $\mathbf{U}_2$, $\mathbf{U}_3$, and $\mathbf{U}_4$.

The intersection algorithm is based on the following idea: if there are no vertices in the interior of $\mathbf{R}$, then it is possible to explicitly determine the shapes of the rubber band curves rbc($\mathbf{AB}$) and rbc($\mathbf{CD}$) inside $\mathbf{R}$. Unfortunately, there may be vertices inside $\mathbf{R}$. To get around this problem, we will create a temporary database which ignores these points.

We create a constant-sized temporary database SIDE′,

$$\mathrm{SIDE}' : \{\mathbf{E}_a, \mathbf{E}_b, \mathbf{P}_i, \mathbf{P}_{i+1}, \mathbf{Q}_j, \mathbf{Q}_{j+1}, \mathbf{U}_1, \mathbf{U}_2, \mathbf{U}_3, \mathbf{U}_4\} \times \{\mathbf{AB}, \mathbf{CD}\} \rightarrow \{\mathrm{ABOVE}, \mathrm{BELOW}, \mathrm{ON}\}.$$

Initially, SIDE′ agrees with SIDE on $\{\mathbf{E}_a, \mathbf{E}_b, \mathbf{P}_i, \mathbf{P}_{i+1}, \mathbf{Q}_j, \mathbf{Q}_{j+1}\} \times \{\mathbf{AB}, \mathbf{CD}\}$. Otherwise, we choose values that are xy-consistent and numerically consistent. Next, we change as many values to ON as possible: for any vertex $\mathbf{P}$ in the domain of SIDE′,

$$\text{if } \mathrm{SIDE}'(\mathbf{P}, \mathbf{AB}) = \left\{ \begin{array}{c} \mathrm{ABOVE} \\ \mathrm{BELOW} \end{array} \right\} \text{ and } \delta(\mathbf{P}, \mathbf{AB})_{\mathrm{R}} \left\{ \begin{array}{c} \leq 4\alpha \\ \geq -4\alpha \end{array} \right\} \text{ then set } \mathrm{SIDE}'(\mathbf{P}, \mathbf{AB}) = \mathrm{ON}.$$

Do the same thing for $\mathbf{CD}$.

It may happen that there are $\mathbf{P}$ and $\mathbf{Q}$ in the domain of SIDE′ such that $\mathbf{PQ}$ is a subset of the boundary of $\mathbf{R}$ and such that

$$\mathrm{SIDE}'(\mathbf{P}, \mathbf{AB}) = \mathrm{BELOW} \qquad \text{and} \qquad \mathrm{SIDE}'(\mathbf{Q}, \mathbf{AB}) = \mathrm{ABOVE}.$$

In this case we simply compute the intersection of this segment with $\mathbf{AB}$ using rounded arithmetic and insert the new vertex into the SIDE′ database.

If after we do all this, there is some point $\mathbf{I} \in \mathbf{R}$ in the domain SIDE′ such that

$$\text{SIDE}'(\mathbf{I},\mathbf{AB}) = \text{SIDE}'(\mathbf{I},\mathbf{CD}) = \text{ON},$$

then **I** is the desired intersection. Otherwise, there will be a subsegment **A′B′** of **AB** which intersects the interior of **R**. In this case, the desired vertex **I** is the intersection of **A′B′** (see Figure 5) with **CD** as calculated using rounded arithmetic. This vertex will have accuracy 5α (Definition 4).

4.6 Updating SIDE

The previous section described how to generate a vertex **I** such that

$$\text{SIDE}'(\mathbf{I},\mathbf{AB}) = \text{SIDE}'(\mathbf{I},\mathbf{CD}) = \text{ON}.$$

Unfortunately, the SIDE′ database was constructed ignoring any vertices lying inside **R**. What do we do if there is a vertex **P** such that SIDE(**P**, **AB**) = ON and $\mathbf{I}_x \leq \mathbf{P}_x$ and $\mathbf{I}_y \geq \mathbf{P}_y$ (see Definition 6)? In this case we cannot set SIDE(**I**, **AB**) equal to ON. What we do is that we throw away **I** and set SIDE(**P**, **AB**) equal to ON.

If we are forced to throw away **I**, we do not add to the number of vertices. We also convert an ABOVE to an ON or a BELOW to an ON, which can happen only a finite number of times. Thus if we keep trying to generate intersections, eventually we will succeed in generating an **I** that works.

5 Conclusion

In conclusion, we would like to answer some questions about the algorithm described above. What sort of applications does it have? Can its complexity be decreased? Are there other approaches?

5.1 Applications

The main application of the segment algorithm is polygon modeling. We define a polygonal region in the plane by listing the set of line segments that form its boundary. If wish to take a union, intersection, or difference of two polygonal regions, the first step is to apply the segment algorithm to the sets of segments. Given the completed SIDE database, generating the desired regions is purely a symbolic, not numerical, process.

There are many applications of polygon modeling in vision and robotics. For example, given a polyhedral scene, we can generate its projection into the image plane using polygon modeling. If we have extracted a region from an image, we can compare it to the projection of a hypothesized model by applying the symmetric set difference. In the field of robotics, we can represent the map used by a mobile robot as a union of polygonal regions. Determining if the robot can fit in a certain location is a matter of taking a polygon intersection. Many other applications surely exist.

5.2 Complexity

The segment algorithm is somewhat complicated. However, if a developer attempts to naively use floating point to solve a geometric problem, he dooms himself to a large period of testing and patching. His final product will be more complicated and less reliable. Correctness more than makes up for the extra complexity.

Rubber band curves are not the only way to approximate segments, and the hidden variable method is not the only way to create robust programs. The real lesson is that there are techniques for creating rounded arithmetic geometry programs that are theoretically well founded. One can also use exact arithmetic even though it appears to be very expensive. Karasick, Lieber and Nackman [8] have recently demonstrated an exact Delaunay triangulation algorithm with empirical cost well below the worst case. In any case, a considerable amount of time and effort can be saved by the small investment of time it takes to investigate these possibilities.

5.3 Acknowledgments

The author thanks Takeo Kanade for his comments, suggestions and guidance on both style and content. Thanks also to Barbara A. Jones for her corrections and suggestions.

References

[1] Jürgen Bokowski, Jürgen Richter, and Bernd Sturmfels. Nonrealizability proofs in computational geometry. *Journal of Discrete and Computational Geometry*, 1988, (to appear).

[2] Herbert Edelsbrunner and Ernst Peter Mucke. *Simulation of Simplicity: A Technique to Cope with Degenerate Cases in Geometric Algorithms.* Technical Report UIUCDCS-R-87-1393, Department of Computer Science, University of Illinois at Urbana-Champaign, Urbana, Illinois, December 1987.

[3] Daniel H. Greene and F. Frances Yao. Finite-resolution computational geometry. In *27th Annual Symposium on the Foundations of Computer Science*, pages 143–152, IEEE, October 1986.

[4] C. Hoffmann and J. Hopcroft. Towards implementing robust geometric computations. In *Proceedings of the Symposium on Computational Geometry*, ACM, 1988.

[5] Christoph M. Hoffmann. *The Problem of Accuracy and Robustness in Geometric Computation.* Technical Report CSD-TR-771, Computer Sciences Department, Purdue University, April 1988.

[6] Christoph M. Hoffmann, John E. Hopcroft, and Michael S. Karasick. *Robust Set Operations on Polyhedral Solids.* Technical Report 87-875, Department of Computer Science, Cornell University, Ithaca, New York 14853-7501, October 1987.

[7] Michael Karasick. *On the Representation and Manipulation of Rigid Solids.* PhD thesis, McGill University, August 1988.

[8] M. Karsick, D. Lieber, and L. Nackman. *Efficient Delaunay Triangulation using Rational Arithmetic.* IBM Research Report RC 14455, IBM Thomas J. Watson Research Center, P.O. Box 218, Yorktown Heights, NY 10598, To appear in 1989.

[9] Victor Milenkovic. Verifiable implementations of geometric algorithms using finite precision arithmetic. *Artificial Intelligence*, 37:377–401, 1988.

[10] Victor J. Milenkovic. *Verifiable Implementations of Geometric Algorithms using Finite Precision Arithmetic.* Technical Report CMU-CS-88-168, Department of Computer Science, Carnegie Mellon University, Pittsburgh, PA 15213, July 1988.

[11] Thomas Ottmann, Gerald Thiemt, and Christian Ullrich. Numerical stability of geometric algorithms. In *Proceedings of the Symposium on Computational Geometry*, ACM, 1987.

[12] D. Salesin, J. Stolfi, and L. Guibas. Epsilon geometry: building robust algorithms from imprecise computations. In *Proceedings of the Symposium on Computational Geometry*, ACM, 1989.

[13] Mark Segal and Carlo H. Sequin. Consistent calculations for solids modeling. In *Proceedings of the Symposium on Computational Geometry*, pages 29–37, ACM, June 1985.

[14] Mark Segal and Carlo H. Sequin. Partitioning polyhedral objects into non-intersecting parts. *IEEE Computer Graphics and Applications*, 8(1), January 1988.

[15] C. Yap. A geometric consistency theorem for a symbolic perturbation theorem. In *Proceedings of the Symposium on Computational Geometry*, ACM, 1988.

RECOGNIZING OBJECTS IN A NATURAL ENVIRONMENT: A CONTEXTUAL VISION SYSTEM (CVS)

Martin A. Fischler and Thomas M. Strat
Artificial Intelligence Center
SRI International
333 Ravenswood Avenue
Menlo Park, California 94025

Abstract

Existing machine vision techniques are not competent to reliably recognize objects in unconstrained views of natural scenes. In this paper we identify a number of weaknesses in current recognition systems, including an inability to solve the partitioning problem or to effectively use context and other types of knowledge beyond that of immediate object appearance. We propose specific mechanisms for dealing with some of these problems and describe the design of a vision system that incorporates these new mechanisms. The system has been partially implemented and we include some experimental results indicative of its operation and performance.[1]

1 Introduction

Most existing work in robotic vision has focused on issues directly related to the immediate geometry and photometry of the scene and the imaging process, such as recovering the three-dimensional location and orientation of scene surfaces from image data and recognizing objects by their geometric shape or by the presence or absence of some prespecified collection of easily measured attributes (e.g., spectral reflectance, texture, and area). On the other hand, most real-world objects are assigned names based largely on their origin, function, purpose, or context. For example, we recognize an object as a bridge not because it has a specific shape, intensity, or texture, but rather because it links the two sections of a road interrupted by an intervening body of water.

This research effort is concerned with the design of a perception system that not only recovers shape but also achieves a physical and semantic understanding of a scene. Such a development would form a link between the low-level, quantitative geometric information currently produced by vision systems and the stylized, symbolic representations that are the mainstay of formal reasoning and planning programs. Until this competence is available, it will remain impossible to develop robots that react intelligently and flexibly to natural environments.

Completely duplicating the human ability to recognize objects is probably equivalent to duplicating human intelligence. An intermediate goal would be to recognize the objects that can occur in completely natural settings (i.e., no human artifacts). Aside from only having to deal with a restricted class of objects and phenomena (probably fewer than 1000 distinct entities), purpose and intent are reduced to recognizing the requirements of vegetation for light, nourishment, and space, as well as the constraints physical forces impose on both living objects and terrain formations. This particular recognition domain derives special interest and importance from the fact that all living creatures must contend with this world and, at least in part, solve this particular recognition problem. Since human recognition abilities evolved in this context, it is reasonable to assume that this domain is extendable (an attribute missing from most other limited "worlds" chosen in the past for the exploration of machine recognition).

[1] Supported by the Defense Mapping Agency and the Defense Advanced Research Projects Agency under contracts MDA903-86-C-0084 and DACA76-85-C-0004

How difficult is the limited problem of recognition in *the natural outdoor world*? *It is safe to say that* few, if any, of the relevant objects can be reliably recognized with existing approaches; even worse, there currently is no credible story about how to proceed. For example, how do we recognize rocks in photographs, what criteria/procedure can be offered to a person who has never seen a rock before (other than offering the advice that after everything else is labeled, the remaining objects are rocks)? What is it that allows us to distinguish a river from a lake, a puddle, or a "run-off" channel, or standing water on a leaf, even after we have locally identified the visible surfaces in a photo that correspond to water?

The investigation we are conducting has a very practical ultimate goal: to enable autonomous vehicles to carry out meaningful work. The research itself has four components:

- To develop an approach that addresses the fundamental limitations of today's systems and that could serve as an architecture for a computational vision system that achieves both a geometric and a semantic understanding of its environment.
- To choose a *vocabulary for describing* the outdoor world. This must include representations for describing a wide variety of shapes and physical attributes, as well as constructs for expressing semantic properties and relations among objects.
- To develop a system that can recognize instances of this vocabulary in imagery from an outdoor domain. This will involve building a knowledge base and suitable control structures to enable the system to recognize natural objects in that domain.
- To demonstrate the feasibility of the approach by implementing a significant portion of the proposed architecture and performing experiments on representative imagery in the context of an autonomous robot.

Our intent in this paper is to frame the recognition problem for natural scenes, offer a reasonable story about how it can be solved, and describe the progress we have made in building a vision system for this purpose.

2 Framing the Problem

What is it that we want to recognize? What *new and prior information* do we require (e.g., with respect to sensory input - a single photo, a sequence of photos, stereo or range information, passive or active sensing, etc.)?

We almost invariably recognize things based on partial information. For example, in a photo, we never see the "backs" of the objects we recognize, but rather assume that they are there. If we recognize a tree in a photo, we might actually see the trunk, or some branches and leaves, but typically (e.g., in a forest scene), even if most of the "parts" are visible, we couldn't correctly assign all the leaves to the correct trunk. Thus, even though we know that trees have trunks, branches, leaves, and roots, it is generally their parts that we recognize and can localize; we then deduce the presence of the whole. It is quite possible that some of the reasonably intelligent animals that live in forested terrain do not partition the world the way we do, and may not even have a concept corresponding to the coherent entity we call a tree. Thus, questions we must address include: what is the vocabulary of objects we want to be able to recognize; and, to what extent should we be able to delineate a recognized object?

What is it that we extract from image data that allows us to recognize natural objects? What knowledge must we have in advance? Edges, contours, color, and texture are far from sufficient cues for recognizing a river, and it is not clear what it is that allows us to recognize a rock. In a sense, rivers are shaped and located by their environment, and it is difficult, if not impossible, to recognize them out of context – i.e., without simultaneously recognizing both the geometry of the surrounding terrain and the co-occurring natural objects. Trees, on the other hand, are shaped by a genetic "blueprint"; they satisfy a set of constraints that are imposed on, rather than derived from, their surroundings. One can probably recognize a line drawing of a tree in isolation, but not of a river. Rocks are intermediate between rivers and trees. They can often be recognized in isolation, but not from a line drawing. They are positioned by their immediate environment, but not shaped by it; they do not conform to a genetic blueprint, but their appearance must reflect a rather narrow set of formative conditions [10].

As a means of extracting a manageable subproblem for our initial effort, the implemented system is intended to recognize a variety of scene components, but will primarily focus on the task of recognizing trees. Given an image of terrain obtained by an autonomous vehicle, and a crude map of the area, the system attempts to identify the obvious trees. We wish to focus on recognition, and will not be unduly concerned with positional accuracy. Objects that are mislabeled are the primary errors that we wish to avoid – mislabelings are the type of "ugly" mistakes that plague today's vision systems.

3 Ingredients of a Solution

The realization of robust recognition in the natural outdoor world will require that four current limitations of machine vision be overcome: the almost exclusive reliance upon shape, the ill-defined nature of the partitioning problem, the lack of an effective way to use context, and the inability to control the growth and complexity of the recognition search space.

The role of geometry – In most existing approaches to machine recognition, the shape of an object or of its parts has been the central issue. Indeed, many artifacts of human technology can be recognized solely on the basis of shape, and to a large degree this fact accounts for the limited success so far achieved by machine recognition systems [2, 4, 12, 13]. These techniques cannot be extended to the natural world because shape alone is insufficient (even for people) to recognize most objects of interest (e.g., a rock or a river). It is easy to recognize a line drawing of an isolated telephone, but, as previously discussed, doubtful that one could correctly classify a river based upon a line drawing of the river alone. Indeed, most natural objects fail this "line drawing test" – a test that requires identification based solely on shape. The fact that very few natural objects have compact shape descriptions further complicates the use of shape in describing natural scenes. Thus a rather complex and cumbersome description would be required to describe the shape of something as common as a tree or a bush. It is obvious that shape cannot be the sole basis for a general-purpose recognition system.

The role of scene partitioning – A common paradigm in machine vision has been to partition an image into distinct regions that are uniform in intensity, texture, or some other easily computed attribute, and then to assign labels to each such region. For natural scenes, however, it is seldom possible to establish complete boundaries between objects of interest. We have already mentioned the difficulty of associating leaves with the correct trees. Other examples are abundant – where does a trunk end and a branch begin, what are the boundaries of a forest, is a partially exposed root part of the ground or the tree? Despite these difficulties, it remains necessary to perform some form of partitioning to do recognition, otherwise we have nothing to refer to when making a classification. Because of the impossibility of performing scene partitioning reliably (even if such a goal were well-defined), we cannot rely on partitioning in the usual sense. Instead, we need an alternative view that allows object recognition without requiring complete or precise object delineation.

The use of contextual information – It is widely known that an object's setting can strongly influence how that object is recognized, what it is recognized as, and if it is recognizable at all. Psychological studies have shown that people cannot understand a scene in the absence of sufficient context, yet when such contextual information is present, recognition is unequivocal [9]. Individual objects may project as a multitude of appearances under different imaging conditions, and many different objects may have the same image. From these studies it is clear that computational vision systems will be unable to classify an object competently using only local information. A perceptual system must have the ability to represent and use non-local information, to access a large store of knowledge about the geometric and physical properties of the world, and to use that information in the course of recognition. However, the few computational vision systems that make use of context do so superficially or in severely restricted ways. In our work, we make the use of contextual information a central issue, and explicitly design a system to identify and use context as an integral part of recognition.

A mechanism for control of complexity – The two standard architectures currently employed in scene analysis are (1) top-down or model-driven interpretation, and (2) bottom-up, a breadth-first form of sensor-driven interpretation. Both of these forms of image analysis (and various intermediate versions) lack an essential attribute of intelligent behavior – an explicit mechanism for generating a (potential or actual) problem solution without requiring some form of exhaustive search. In controlled or simple environments, exhaustive search may be computationally feasible, but the complexity of the natural world imposes the requirement for a more efficient solution mechanism. A key aspect of our approach is the provision of an explicit mechanism for generating high-quality assertions about the scene without the need for exhaustive search.

4 The Shape of a Solution

An effective scene-recognition system must provide:

- A "language" for making semantically meaningful assertions about a scene.
- Computationally effective procedures for generating and evaluating hypotheses (assertions) about scene objects and relations.
- Methods that permit validation of a scene description at a global level.

The key ideas underlying our approach are:

1. The selection of the outdoor navigation and mapping tasks as the basis for delimiting the semantic knowledge the system must be able to employ in describing and understanding a scene (some relevant vocabulary items are listed in Table 1).

geometric horizon	sky	ground
skyline	thin vertical raised object	occluding edge
raised object	foliage	tree trunk
tree crown	bush	tree
cloud	ridge line	branch

Table 1: Vocabulary of terms

2. Explicitly encoding the different sets of conditions (contexts) under which the semantic objects we are interested in can be reliably compared and recognized. These recognition conditions, which typically require the prior or co-determination of the presence of other semantic features, are called *context sets.* Some of the criteria that comprise context sets are listed in Table 2. High-level knowledge must be brought to bear if recognition is to be performed competently. As we will see, context sets are the central knowledge source around which our recognition system is constructed.

GLOBAL CONTEXT:	
	foothills on San Francisco peninsula
	daytime
	cloudless sky
LOCATION:	
	local topography
	touching the ground
	coincident with a known tree
APPEARANCE:	
	geometric: shape, size, neighbors, depth, orientation
	photometric: intensity, texture, color
FUNCTIONALITY:	
	supporting another object
	bridging a stream
	counterbalancing a tree limb

Table 2: Examples of criteria that comprise context sets

3. The design of a control structure to limit the computational complexity of the recognition process: (a) control of the quality of hypothesis generation through a biased form of image partitioning, i.e., the use of region delineation based on a context-set-adjusted homogeneity metric tailored to find portions of an image associated with specific scene features; and (b) control of search (in recognition space) by creating partial orderings of all of the hypotheses making assertions about the presence of a particular type of scene feature.

4. Recognition is accomplished by finding mutually consistent groupings (*cliques*) of hypotheses about the different semantic features. Cliques are formed by sequentially adding the highest ranking hypotheses (with respect to the partial orderings) that are consistent with the current global interpretation.

5. Physical and imaging constraints, in the context of a 3-D spatial model, are employed as the primary criteria for the consistency of a global interpretation (rather than probabilities for different arrangements of semantic objects).

4.1 Scene Semantics

There are probably on the order of 100 to 1000 semantic object classes that an organism (such as a rabbit or a robot) needs to be able to identify in order to move safely about in a relatively benign environment (such as the grass- and tree-covered rolling hills in the San Francisco Bay area), but for our initial experiments we will focus on finding trees and will recognize other scene features only to the extent of supporting this primary objective (these additional scene features include sky, ground, thin raised objects, shadows, occlusion edges, ...). As indicated earlier, many of the naively chosen object classes we might deem appropriate for describing a natural scene require complex reasoning about those parts of the object that are directly observable; thus, in our first experiments, we will actually look for vertically oriented tree trunks rather than actual trees. The determination of a semantic vocabulary that matches the recognition capabilities of a given vision system and satisfies the requirements of a specified task is a non-trivial issue that we are investigating but will not further address in this paper.

4.2 Context Sets

Computer vision presents the following paradox: For natural scenes, in order to recognize an object, its surroundings must first be recognized, but in order to recognize the surroundings the scene objects must often be recognized first. Is it really necessary to recognize everything at once, or can some things be recognized in relative isolation? If so, what are they, and how can they be recognized? It is certainly the case that one does not have to know about everything in the universe to recognize (say) a tree; on the other hand, trying to find the trees in an image by looking through a small "peep-hole" may be impossible. A critical aspect of our approach is to define explicitly sets of conditions (context sets) under which various semantic objects (such as trees) can be recognized – or at least ranked (in terms of their "tree-ness"). Typical conditions employed in context sets are presented in Table 2. Our approach makes use of several forms of context:

- *Global context* – Trees differ greatly from one climatic zone to another. Knowing whether the image is of an arctic or tropical environment should affect perceptual strategies.
- *Location* – An object flying through the air is more likely a bird than a rabbit – and is almost certainly not a tree. These facts can be deduced from context alone, without any reference to an object's shape, color, or texture.
- *Appearance* – Once a tree has been recognized, it is easy to recognize others, because neighboring trees are often similar in appearance.
- *Functionality* – A tree trunk that has fallen across a stream can act as a bridge, but can be identified as a bridge only after recognition of the stream.

There is a collection of context sets for each semantic category in the vocabulary and each context set is associated with exactly one category. The system employs three kinds of context sets:

(1) A *hypothesis generation context set* is used to generate candidate hypotheses for the category it is associated with. A hypothesis is an assertion that, for example, a certain specific region in an image corresponds to a tree trunk. Each context set contains the conditions under which a computational process (*operator*) should be employed and supplies the particular parameter settings that should be used in that context. The operator is only invoked when the conditions of the context set are satisfied. All hypothesis generation context sets whose conditions are satsified are used to generate candidates.

(2) A *hypothesis validation context set* is used to rule out certain candidate hypotheses from further consideration as instances of a class. It specifies a set of conditions such that if the conditions are not satisfied, the hypothesis could not denote a member of the class. For example, an object whose diameter is greater than fifty feet could not be a tree trunk. These context sets eliminate the obviously erroneous hypotheses that might have slipped through the earlier processing.

(3) A *hypothesis ordering context set* is used to rank order a pair of candidate hypotheses. It contains conditions under which one hypothesis should be preferred over another as an instance of the class associated with the context set. If any hypothesis ordering context set asserts a preference between two hypotheses, that preference is recorded. Otherwise, no preference is established. Pairwise comparison of all candidate hypotheses for a class results in a partial ordering of hypotheses for that class.

4.3 Control of Complexity: Hypothesis Generation and Ordering

Perception is appropriately considered a form of intelligent behavior because it nominally involves searching an infinite "description space" to find an appropriate model of some imaged environment that conceivably could have any one of an infinite number of arrangements of its components. Most current vision systems avoid addressing this central problem by restricting the dimensionality of the relevant search space, for example, by dealing exclusively with controlled environments where only a few, completely modeled, objects can occur. That is, we have complete parameterized descriptions of the environment in advance; our task is to find appropriate values for a relatively small number of numeric parameters. Because such systems have no provision for handling the combinatorics of complex natural environments, they are not capable of "scaling up" to real-world scenes.

It is apparent that a central consideration in the design of a "real world" vision system is the development of some mechanism for controlling the quality (and thus the quantity) of the assertions/hypotheses that must be filtered by the system. We deal with this problem by the use of prior knowledge, by context-set-controlled partitioning, and by quality-based ordering of the generated hypothesis for preferential use in constructing global descriptions.

4.3.1 The Use of Prior Knowledge

We envision the system we are constructing as being part of an autonomous robot situated in the world, so a great deal of information will be available to it. Furthermore, the spatial and temporal continuity of the world leads to added opportunities for a robot vision system to improve its performance based on its experience.

The following facts are known by reasonably intelligent animals and are expected to be immediately available to a robot as well. The approach we are taking is designed to make use of these items of information:

- *Height of viewer above the ground:* An animal knows its orientation intrinsically because its eyes are fixed in relation to a given configuration of its body (assuming its feet are on the ground). This value is fixed (or easily computed) for a robot as well.
- *Orientation of the viewer (with respect to gravity):* An animal knows this by virtue of its sense of balance in the inner ear. A robot could know this by employing a simple sensor.
- *Vertical vanishing point:* Knowing the orientation of the viewer with respect to gravity allows one to compute the vertical vanishing point in the image plane.
- *A depth image:* Through stereopsis, an animal can estimate the distance to the things that it sees, although the quality, density of coverage, and accuracy are debatable, and the precision available to it degrades with distance. A robot might employ several means to acquire depth data. Binocular stereo techniques provide depths with various qualities. Laser rangefinders provide dense depth images but have other limitations. Other techniques (motion, texture, shading) and sensors (acoustic, structured light) are also available.
- *The ground (at least one point in an image that can be labeled as ground):* An animal can identify at least one point that is known to be on the ground by looking at its feet (assuming the animal is currently standing on the ground). A robot could perform the same action to locate a ground point.
- *The sky (at least one point in an image that can be labeled as sky):* An animal can find at least one point that is known to be sky by looking straight up (assuming it is not currently underneath a tree, in a cave, etc.) A robot could do the same.

- *Experience:* Unless the terrain is totally new, an animal has knowledge of the relative locations and appearance of some features and can predict the appearance of parts of a scene. While the vocabulary and representation of this information remain research issues, a robot should store and use this type of information as well.

4.3.2 Context-Set-Controlled Partitioning and Hypothesis Ordering

One of the unusual aspects of our approach is its organization. Over the years, many different strategies have been proposed for extracting information from images. It has become apparent that no single general-purpose strategy will suffice to address the many requirements placed on vision systems. Our approach is designed to choose low-level operations based on the semantics and context of each visual task. Rather than applying a fixed operator in all circumstances, we make strong use of semantics to determine the most appropriate low-level processing.

The underlying structure of the design is motivated by the following key observation:

> Within any given image, there is usually a relatively straightforward technique that will find the object or feature of interest.

Of course, that particular technique may fail miserably when applied to some other image. However, given a collection of simple techniques, one can generate a number of candidate features that will nearly always contain the "correct" interpretation. It then remains to choose that feature from among all the candidates. To accomplish this, we make use of a novel scheme:

> First, we bias the feature (hypothesis) generation process to be more responsive to generating good hypotheses relevant to a specific semantic category. Then we compare the candidates pairwise to find instances that are clearly "better" examples of the given class than their opponents. Repeating this comparison process imposes a partial order on the candidate set for each label of interest.

This departs from conventional approaches in two ways. First, comparing two candidates for a given label requires knowledge of the semantics of that label only, whereas the customary approach of comparing two labels for a given region requires knowledge of the relationships between many semantic categories. This orientation provides a basis for believing that sufficient knowledge might be encoded in the system to allow robust comparison. Second, we enforce the condition that the comparisons lead to a determination only if one candidate is clearly a better choice than the other. With this conservative approach, we hope to avoid the "ugly mistake" of misclassifying a candidate based on too little information.

The actual hypotheses generated by the system are motivated and shaped by the context sets in a manner somewhat analogous to a collection of production rules. Separate partitions of an image are created for each relevant semantic object class (e.g., sky, ground, trees ...), and in some cases, for each specific context set. The basic mechanism for creating a partition is an operator that assigns to each pixel in the image a homogeneity score which is a measure of its maximum "feature space distance" from any of its immediate neighbors. The components of the feature space, and their relative weightings, are specified by the currently active context set.

The output of the homogeneity operator is a gray-scale overlay of the original image that can be thresholded (under context set control) to form a binary image in which the "zero points" either are discontinuities or correspond to some feature other than the one associated with the context set; the "one points" are accumulated into coherent regions that are indicative of the presence of the feature of interest (see Figures

1 - 4). The context set may require that other generic operations be performed on the detected regions; e.g., "skeleton" generation as required for tree detection.

In actual practice, the context sets have varying degrees of specificity and precision. The more general and low-resolution context sets will operate first (because their conditions are more easily satisfied), and they typically use the output of the generic image operators with some standard set of parameter values. Thus, it is only necessary to run a few generic operators over the image once in forming an initial scene model. (Table 3 describes the collection of generic operators we currently employ.) Specialized operators and generic operators with customized settings are typically required by the context sets that can only be invoked after the initial model is formed - these context sets make more precise assertions about the scene, or deal with known exceptions to the general rules for finding the relevant object instances. For example, the system might require a special context set to recognize a lightning-struck tree that is known to be present, but no longer looks like a normal tree.

TEXTURE	- Measures the log-variance in a small window
STRIATION	- Measures the orientation and strength of an oriented pattern
THRESHOLD	- Creates a binary mask
SEGMENTATION	- Partitions an image
ASSOCIATION	- Groups contiguous clusters in a binary mask
EDGE DETECTION	- Finds discontinuities
LINEAR DELINEATION	- Finds line-like structure
HOMOGENEITY	- Measures the maximum feature-space difference between pixels in a small neighborhood

Table 3: Operators

The system constructs a global scene model by incrementally assembling mutually consistent collections of hypotheses (generated by the process just described). The hypothesis generation context sets are used to assure that only reasonable assertions are made; the hypothesis ordering context sets are used to provide an ordering for selecting the most likely hypotheses to add to the model. There is a collection of context sets for each class of objects the system is to recognize. If any one of them has a preference during a comparison of two hypotheses, then that preference is taken into account by incorporating it in the partial order. If no context set can establish a preference, or if several context sets disagree (which ideally should never happen and would require a modification of the context sets involved), then no preference is recorded for that comparison. In this way, a partial order is constructed from all hypotheses for each object class.

4.4 Constructing a Global Scene Model

Once partial orders have been constructed for the labels of interest, and for the additional labels that appear as terms in the primary context sets, a search for a consistent interpretation of the scene is conducted. Hypotheses are individually added to cliques, checking for consistency with those already present in the clique. The search begins with those candidates at the tops of the partial orders and progresses until no more hypotheses can be added without causing a contradiction. The clique that explains the largest portion of the image is offered as the best interpretation. The result should represent a reasonable explanation of major portions of the image.

4.5 Model Consistency

Determining the "correctness" of a scene model is analogous to establishing the validity of a scientific theory – one can never be assured that the model is valid: the best we can do is to filter out incorrect models by showing they have some internal contradiction or violate some accepted fact. In the approach we have described for constructing a scene model, the problem of assuring global consistency is addressed by the mechanism for clique formation – the addition of a new assertion about the scene should not be able to cause the satisfaction of the conditions of other context sets that cause rejection of already accepted hypotheses.

Assertions about the different semantic categories constrain each other in an absolute way in terms of establishing 3-D geometric and physical relationships that must be consistent with known (a priori) facts about the environment and the object categories (constraints on size; support; orientation; occupancy of solid objects; and free-space imposed by the viewing geometry of the image). For example, if an object was identified as a tree at some unknown distance, and a later assertion established the distance to the tree so as to allow us to check its dimensions against known generic tree size limits, we might find that a contradiction exists between the distance assertion and the tree identity assertion. Thus, if an object is identified as a tree, then a region of the image completely occluding a portion of the trunk certainly cannot be sky. Even if the system has enough knowledge to deduce this fact through activation of appropriate context sets (i.e., (1) an occluding object is closer than the object it occludes, (2) the sky is infinitely far away, (3) if the sky occludes the tree trunk, the tree trunk is infinitely far away, (4) a visible object at infinite distance has infinite size, (5) trees have finite size), explicit knowledge of this contradiction might remain unknown to the system. Since there is no explicit mechanism for accomplishing the above reasoning, the contradiction would only be recognized if we explicitly force the propagation of all distance and size assertions (i.e., make explicit all physical assertions). Rather than attempting to achieve this purpose by random or exhaustive expansion of our knowledge base, we build a (simulated) analogical model of the environment as a way of *directing the deductive process in evaluating global consistency.* Thus, consistency checking is accomplished by 3-D model construction according to a set of conditions that prevent a non-physically-realizable situation from occuring. The resulting 3-D model is one of the primary outputs of the system.

5 Progress

The adequacy of our approach is largely an empirical question which we address experimentally using real imagery. The implementation (called CVS, for Contextual Vision System) is not complete, but enough has been put in place to provide some preliminary results. The system is built on a number of other components that we routinely use in our work. One of these, the Core Knowledge System (CKS), provides the primary representation and storage mechanism for the actual and hypothesized scene entities derived by the CVS [19, 20].

Our implementation strategy has been first to construct a control structure to carry out all phases of the approach. This has been completed. Second, an instantiation of the knowledge base has been accomplished for a thin slice of the knowledge that must be present in a fully functional system. We have used results obtained from this partial system to guide the design of the remainder of the knowledge base and to provide insight into the merits and limitations of our approach. Constructing the knowledge bases for a perceptual system can be a tedious and difficult task. It is clear that some form of automated knowledge acquisition (learning) is desirable and perhaps necessary. We are exploring ways to add such a facility to CVS.

5.1 Vocabulary

As mentioned earlier, the choice of vocabulary is crucial, and the best terms to include in the vocabulary may not be the most obvious ones. The ıerms we currently employ for the recognition of trees were listed in

Table 1. They were chosen on the basis of three factors: (1) they can often be recognized in relative isolation - knowledge of the presence of other classes is probably not necessary for finding potential candidates; (2) they appear to be useful for setting the context for the recognition of other objects; and (3) operators can be constructed to reliably locate candidate instances. As the system matures, we will introduce new terms into the vocabulary.

5.2 Control of Complexity

In the customary machine vision paradigm, an image is partitioned into disjoint regions, and a unique label is assigned to each one. If there are r regions and k labels, then r^k possible labelings exist. Various constraints and heuristics are employed to find the best labeling within this search space, which is exponential in the number of potential labels.

Within CVS, each operator generates a small number of candidate regions, which are then ordered by the applicable context sets and added to cliques. Suppose a correct interpretation of an image contains r regions. Let p be the probability that a region that is about to be added to a clique is part of a consistent interpretation. The probability of adding r such regions to a single clique is p^r. On average, one would have to try constructing p^{-r} cliques before a valid one with r regions is obtained. Since each clique requires up to r regions, the complexity of clique construction is $O(rp^{-r})$. Thus, any contextual recognition scheme, such as ours, is potentially exponential in the number of regions in the result. A coarse description is generated rather quickly - more detail can be added but at an exponentially increasing cost. The key to attaining a manageable level of complexity is the ability to offer a candidate region to a clique with a high probability of acceptance. If $p = 1$, then the complexity is $O(r)$. Obviously, the closer that p approaches 1, the longer one can ward off a combinatoric explosion. The quality of the candidate generators certainly affects p - having fewer incorrect candidates yields a higher probability of acceptance. The process of partial order construction was designed to boost p even higher. By ensuring that the best examples of a label are introduced first, one may avoid the need to introduce some of the less likely candidates.

5.3 Hypothesis Generators

A hypothesis generator is implemented as a combination of low-level operations that delineates in an image one or more regions as candidates for a particular vocabulary term. Our strategy has largely been to employ standard image-processing routines that have been developed by the vision community through the years. These are combined in various ways to tailor their output according to the specifications of the context sets. Some of the operators we currently employ were listed in Table 3.

5.4 Hypothesis Ordering

Partial orders among candidate regions are created by pairwise comparison of candidates. Context sets associated with each vocabulary term are employed to perform the pairwise comparison. A context set defines a collection of related criteria that is sufficient to prefer one candidate over another as an instance of its class. Examples of such criteria are listed in Table 2. When some criterion is not satisfied, a context set will offer no preference between two candidates. A preference relation will be added to the partial order if any context set offers a preference. Examples of several partial orderings of candidate regions is shown in Figures 5 - 7.

There is one special "context set" for each semantic category that enforces constraints on the physical properties of the object (e.g. tree trunk width, maximum tree height, maximum branch lengths). If any physical property of an object fails to satisfy a listed constraint, the object cannot be given the corresponding label and is removed from the partial order.

5.5 Global Consistency

A labeled region is not accepted until a mutually consistent set of labeled regions is identified that explains the image features. Cliques of consistent regions are formed incrementally by nominating a candidate for inclusion, evaluating it for consistency with the existing clique, and adding it to the clique. If a nominee is found to be incompatible with a clique, it is removed from its partial order and an alternate nominee is selected. If a nominee is consistent, it is added to the clique, and all context sets are (theoretically) reevaluated in light of the new context represented by the expanded clique.[2]

A 3-D model of the evolving clique is created by inserting object tokens into the Core Knowledge System [19, 20]. The spatial and semantic retrieval mechanisms of the CKS are employed to establish context for further processing by other context sets. The CKS maintains the data associated with each clique as separate opinions so that hypotheses maintained by one clique do not interfere with those of another clique.

Nominees are chosen according to various heuristics that attempt to maximize the chance that a candidate will be accepted into a clique. Currently, we use the simple heuristic of choosing the largest region that is atop any partial order. Later, we intend to allow the context to suggest which candidate from which class should be nominated.

An initial consistency check is performed in the image plane. A nominee is checked for overlap with any region already in the clique. If the overlap exceeds a threshold, the nominee is ruled to be inconsistent. This simple-minded scheme is intended to serve as a place-holder until a module capable of full three-dimensional consistency checking is completed.

5.6 Illustration of Clique Formation

We now present an example showing the preprocessing and result of clique formation. Some aspects of the example have been hand-edited for the sake of perspicuity. Figure 8 portrays an image of some trees on the Stanford campus that has been presented to CVS for interpretation. After applying a number of context sets and their associated hypothesis generators, partial orders for the vocabulary terms sky, foliage, ground, and tree trunk have been created (see Figures 5 - 7).

Suppose that sky candidate number 593 (which was generated by a simple thresholding on intensity) is nominated first. It is added to an empty clique and the corresponding volume is marked as sky in the CKS. Any context sets that might now be satisfied are reevaluated. This reevaluation may cause the generation of new candidates or may change some of the partial orders, but for the remainder of this example, we'll assume that does not happen. Next, foliage candidate 543 is introduced. It overlaps somewhat with the sky region in the clique, but not enough to flag an inconsistency. So 543 is added as foliage; its volume (estimated using range from stereo) is inserted in CKS as foliage, context sets are reevaluated, and processing continues. Next ground region 549 is nominated. Its overlap with the sky region already in the clique is greater than the allowed threshold for ground-sky overlap and is ruled inconsistent. Candidate 549 is removed from the ground partial order and processing continues. Suppose foliage candidate 545 is nominated next. Its overlap with the sky region already in the clique is small enough, so its volume is computed and inserted in the CKS. But, because it is completely contained in the sky volume it has no possibility for support, and is ruled as inconsistent. It and all its inferiors in the foliage partial order are removed from consideration because there are context sets that have already determined that 545 is a better example of foliage than 544 or 594. If 545 is not foliage, than 544 and 594 cannot be foliage either. Further nomination of candidates 554 (sky), 540 and 536 (ground), 546 (foliage) and 537 (tree trunk) yields a clique containing (536 537 540 543 546 593) - its coverage of the image is portrayed in Figure 9(a).

[2] In practice, a "lazy" evaluation scheme is employed to perform the reevaluation efficiently.

Now suppose a new clique is created, but that sky candidate 556 is the first nominated. Foliage candidate 543 is nominated and accepted. When ground candidate 549 is nominated, no overlap with the sky region is found, so 549 is accepted into the clique as ground. Foliage candidate 545 is now found to be supportable by the ground, so it is accepted as well. Further processing results in a clique containing (536 537 540 543 544 545 546 549 556 595). The coverage of this clique is shown in Figure 9(b). Because it explains more of the image than the previous clique, it is accepted as the better interpretation.

6 Concluding Discussion

6.1 Prior Work

Nearly all of the existing work on recognition (by a robotic vision system) has been conducted in a context where precise geometric models of the relevant objects are known beforehand, and the major goal has been to find projections of the various models that best match some part of an image [2, 4, 12, 13]. To relax the requirement for complete and accurate models, Fischler and Elschlager introduced the technique of deformable (spring-loaded) templates [6], which represent objects as a combination of local appearances and desired relations among them (the "springs"). An object represented in this way is located in an image by using a form of dynamic programming to simultaneously minimize local and global evaluation functions. Some geometric recognition systems, such as ACRONYM [3] accept parameterized models to recognize a class of objects, but these too are overly restrictive to be of much use with natural features.

For the natural world, precise geometric models of natural objects are not available, and existing techniques offer little insight on how to proceed. There has been some work directed toward the goal of semantic understanding of natural outdoor scenes [1, 5, 7, 11, 14, 16, 17, 18, 21, 22], but surprisingly, very little new work has been initiated in the last ten years.[3] All of these approaches begin by partitioning the image into regions, which presumably mirrors the "natural" decomposition of the scene into "objects." The regions are then analyzed in one way or another to determine their interrelationships, to merge them into larger regions, and ultimately, to assign each region a label that categorizes it semantically.

6.2 Our Contribution

Some of the key differences between our work and previous efforts include recognition in the absence of explicit shape models; no reliance on accurately partitioned and delineated objects; no requirement for logically consistent absolute constraints; and no use of probabilistic models requiring a priori probability values and independence assumptions.

The critical issue that must be resolved in formulating a viable control strategy is whether it is necessary to recognize everything at once (e.g., via relaxation), or whether some critical scene elements can/must be recognized first in relative isolation. If so, what are these elements and how can they be found? In a similar sense, what volume of space, context, etc. constitutes a smallest interpretable unit? We take a relatively unique position based on the following assertion: almost nothing in nature can be visually identified in isolation. Therefore, every interpretation rule must have an explicitly stated contextual setting to which its use is restricted.

The ability to generate "good" hypotheses and perform reliable comparisons of candidates for a particular label is one of the most important aspects of our system. We have devised a new mechanism, known as *context sets*, to support these functions. The name derives from the need to compile a set of information

[3] The major exception to this statement is the work sponsored by the DARPA Strategic Computing program. However, much of this work, such as that described in this paper, has still not reached a full stage of maturity.

sufficient for deciding whether one candidate should be preferred over another. This information, then, constitutes the context for recognition in that circumstance. The context sets are the mechanism employed to account for such factors as view angle, scale, and geographic location.

The context sets are the main repository of knowledge within our system. In addition to supporting the ordering of hypotheses, and thus the efficient construction of a global model, they play a major role in hypothesis generation and in checking global consistency - the context sets are thus an integrative mechanism linking all the different knowledge levels the system must be concerned with.

A key idea in the way the system is organized is that we do not make direct decisions that choose between two different class labels for a detected region in an image - the context sets and the partial orderings they create deal only with one semantic category. Thus we avoid the combinatorics of having to (explicitly) describe how to distinguish among combinations of different class labels for an unknown object. We also avoid the need to make major modifications in our knowledge base when some new object type is added. Recognition in the CVS is accomplished when a globally-consistent, labeled 3-D model has been constructed.

7 Acknowledgment

We would like to acknowledge the contributior of Helen Wolf and Lynn Quam, who shared their expertise and assisted with the system implementation and with the experiments described here.

References

[1] Barrow, Harry G., and Tenenbaum, Jay M., "MSYS: A System for Reasoning about Scenes," Technical Note 121, Artificial Intelligence Center, SRI International, April 1976.

[2] Bolles, R.C., Horaud, R., and Hannah, M.J., "3DPO: A 3-D Part Orientation System," *Proceedings 8th International Joint Conference on Artificial Intelligence*, Karlsruhe, West Germany, August 1983, pp. 1116-1120.

[3] Brooks, Rodney A., "Model-Based 3-D Interpretations of 2-D Images," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Volume 5, Number 2, March 1983, pp. 140–150.

[4] Faugeras, O.D., and Hebert, M., "A 3-D Recognition and Positioning Algorithm using Geometrical Matching Between Primitive Surfaces," *Proceedings 8th International Joint Conference on Artificial Intelligence*, Karlsruhe, West Germany, August 1983, pp. 996–1002.

[5] Feldman, Jerome A., and Yakimovsky, Yoram, "Decision Theory and AI: A Semantics-Based Region Analyzer," *Artificial Intelligence*, Vol. 5, No. 4, 1974, pp. 349–371.

[6] Fischler, M. A., and Elschlager, R. A., "The Representation and Matching of Pictorial Structures," *IEEE Transactions on Computers*, Volume C-22, Number 1, January 1973, pp. 67–92.

[7] Fischler, Martin A., Bolles, Robert C., and Smith, Grahame, "Modeling and Using Physical Constraints in Scene Analysis," Technical Note 267, Artificial Intelligence Center, SRI International, September 1982.

[8] Fischler, Martin A., and Wolf, Helen C., "Linear Delineation," Proceedings IEEE Computer Vision and Pattern Recognition, 1983, pp. 351-356.

[9] Fischler, Martin A., and Firschein, O., *Intelligence: The Eye, the Brain, and the Computer*, Addison-Wesley Publishing Co., 1987, pp 220–229.

[10] Fischler, Martin A., and Strat, Thomas M., "Recognizing Trees, Bushes, Rocks and Rivers," *Proceedings of the AAAI Spring Symposium Series: Physical and Biological Approaches to Computational Vision*, Stanford University, March 1988, pp. 62–64.

[11] Garvey, Thomas D., "Perceptual Strategies for Purposive Vision," PhD Thesis, Department of Electrical Engineering, Stanford University, December 1975.

[12] Goad, C., "Special Purpose Automatic Programming for 3D Model-Based Vision," *Proceedings: DARPA Image Understanding Workshop*, June 1983, pp. 94–104.

[13] Grimson, W.E.L., and Lozano-Perez, T., "Model-Based Recognition from Sparse Range or Tactile Data," *International Journal of Robotics Research*, Volume 3, Number 3, 1984, pp. 3–35.

[14] Hanson, A.R., and Riseman, E.M., "VISIONS: A Computer System for Interpreting Scenes," in *Computer Vision Systems*, Academic Press, New York, 1978, pp. 303–333.

[15] Laws, Kenneth, I., "Integrated Split/Merge Image Segmentation," Technical Note 441, Artificial Intelligence Center, SRI International, July 1988.

[16] McKeown, David M., Jr., and Denlinger, Jerry L., "Cooperative Methods for Road Tracking in Aerial Imagery," *Proceedings: DARPA Image Understanding Workshop*, April 1988, pp. 327–341.

[17] Rosenfeld, A., Hummel, R.A., and Zucker, S.W., "Scene Labeling by Relaxation Operations," *IEEE Trans Systems, Man, Cybernetics*, Volume 6, Number 6, June 1976, pp. 420–433.

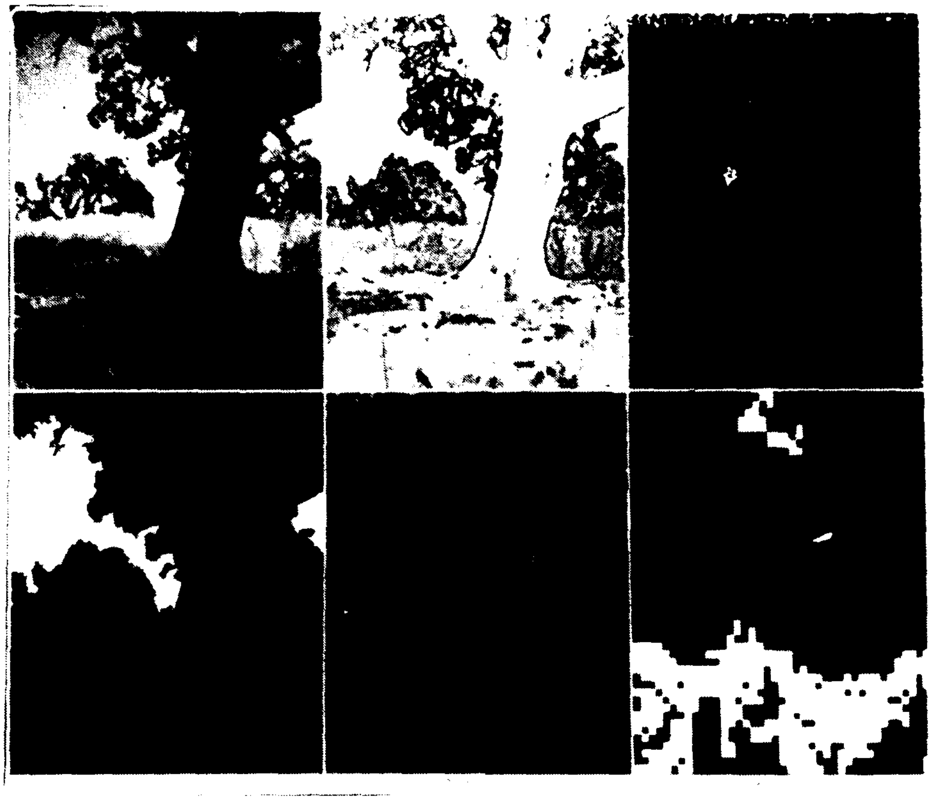
[18] Sloan, Kenneth R., Jr., "World Model Driven Recognition of Natural Scenes," PhD Thesis, Moore School of Electrical Engineering, University of Pennsylvania, Philadelphia, Pennsylvania, June 1977.

[19] Smith, Grahame B., and Strat, Thomas M., "Information Management in a Sensor-Based Autonomous System," *Proceedings: DARPA Image Understanding Workshop*, February 1987, pp. 170–177.

[20] Strat, Thomas M., and Smith, Grahame B., "The Core Knowledge System," Technical Note 426, Artificial Intelligence Center, SRI International, October 1987.

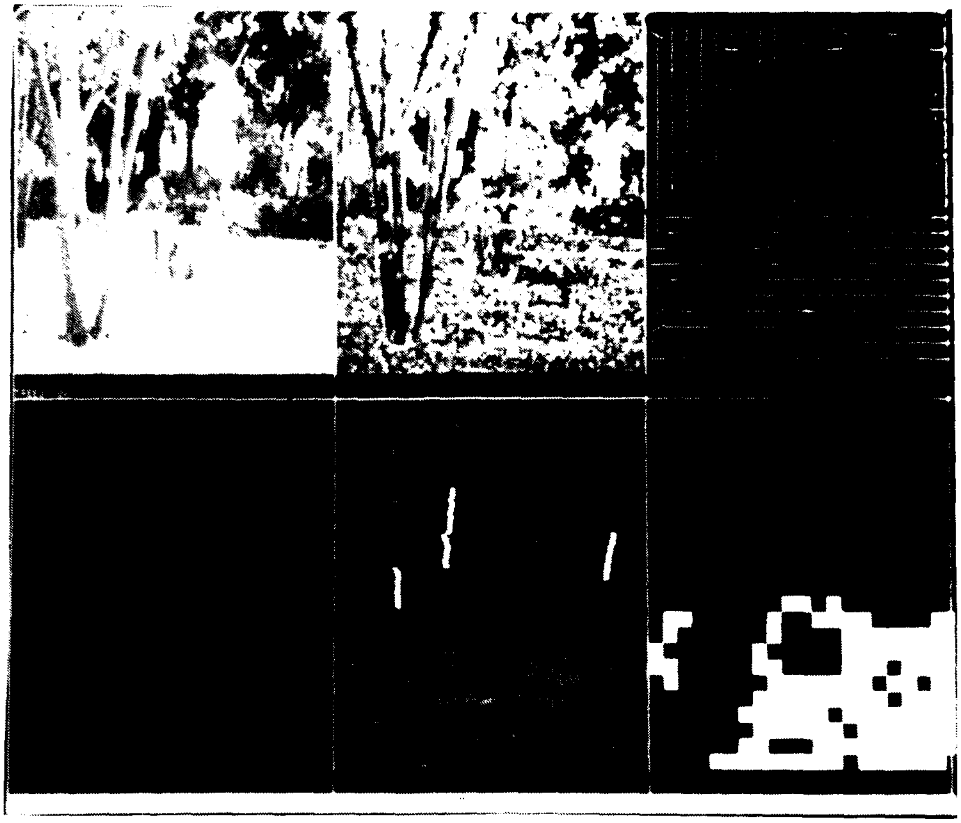
[21] Tenenbaum, J. M. and Weyl, S., "A Region Analysis Subsystem for Interactive Scene Analysis," *Proceedings of the Fourth International Joint Conference on Artificial Intelligence*, September 1975, pp. 682–687.

[22] Yakimovksy, Yoram, and Feldman, Jerome A., "A Semantics-Based Decision Theory Region Analyzer," *Proceeding of the Third Joint Conference on Artificial Intelligence*, August 1973, pp. 580–588.



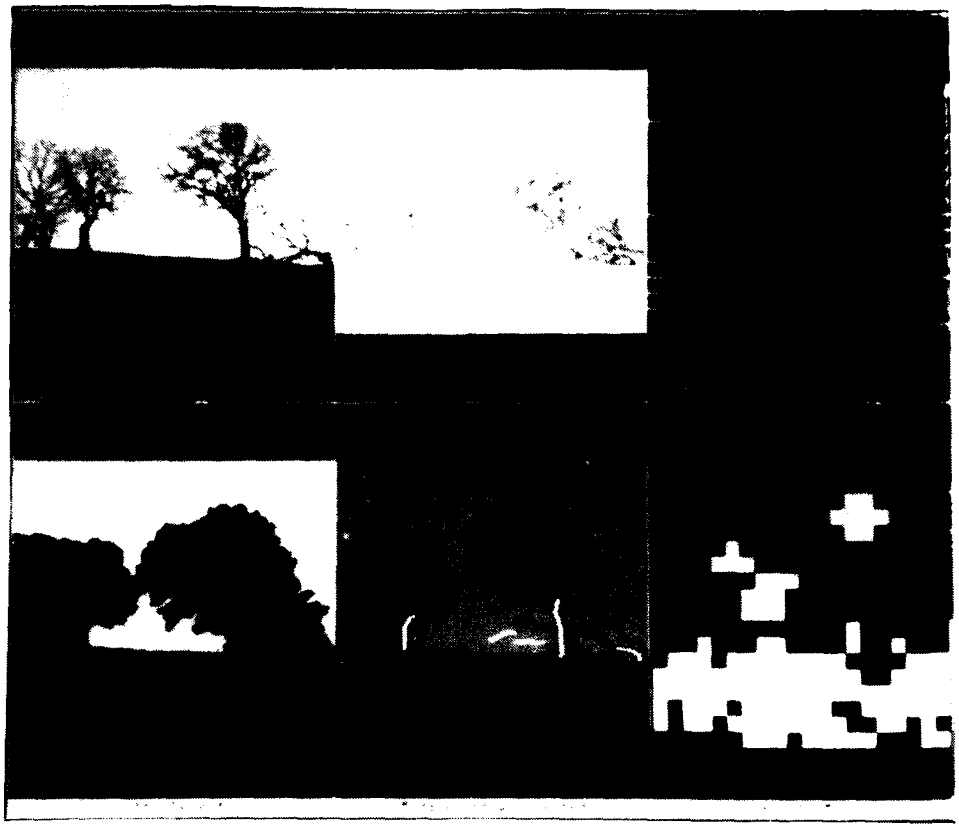
(a) B&W Image of some trees near Stanford	(b) Homogeneity operator - Each pixel value is the maximum difference in intensity between it and all neighboring pixels.	(c) Striations operator – Line segments show the orientation of any texture pattern in a small window.
(d) Sky region hypotheses - The entire scene was partitioned by Laws' segmenter, KNIFE [15]. Each region that is above the geometric horizon, is relatively bright, and is relatively untextured is displayed.	(e) Tree trunk hypotheses - Coherent regions were grown from the output of the homogeneity operator (b) above. Skeletons of the major regions were constructed and filtered to remove short and highly-convoluted skeletons. The tree trunk and its major limbs have been identified (superimposed in white on the original image).	(f) Ground region hypotheses – Regions of horizontal striations were extracted from (c) above. Small regions have been discarded. Horizontal surfaces tend to have horizontal striations when viewed from an oblique angle due to perspective foreshortening.

Figure 1: Output of various operators applied to a natural scene



(a) B&W Image of some trees and a stump	(b) Homogeneity operator – Each pixel value is the maximum difference in intensity between it and all neighboring pixels.	(c) Striations operator - Line segments shcw the orientation of any texture pattern in a small window.
(d) Sky region hypothe.ses - The entire scene was partitioned by the KNIFE segmenter. There were no relatively bright, untextured regions above the geometric horizon.	(e) Thin object hypotheses – A linear delineation operation [8] was performed on the output of the homogeneity operator (b) above. Short segments and highly-convoluted segments were removed. Portions of several tree trunks have been identified.	(f) Ground region hypotheses – Regions of horizontal striations were extracted from (c) above. Small regions have been discarded. Notice how the stump is not included.

Figure 2: Output of various operators applied to a natural scene



(a) B&W Image of some trees near Stanford	(b) Homogeneity operator – Each pixel value is the maximum difference in intensity between it and all neighboring pixels	(c) Striations operator – Line segments show the orientation of any texture pattern in a small window
(d) Sky region hypotheses – The entire scene was partitioned by the KNIFE segmenter. Each region that is above the geometric horizon, is relatively bright, and is relatively untextured is displayed.	(e) Thin object hypotheses – A linear delineation operation was performed on the output of the homogeneity operator (b) above. Short segments and highly-convoluted segments were removed. The trunks of the trees were successfully delineated, although some spurious candidates remain to be filtered out later.	(f) Ground region hypotheses – Regions of horizontal striations were extracted from (c) above. Small regions have been discarded. Some branches of the trees were picked up, but these candidates should be eliminated during later processing.

Figure 3: Output of various operators applied to a natural scene

(a) B&W Image of some trees	(b) Homogeneity operator – Each pixel value is the maximum difference in intensity between it and all neighboring pixels	(c) Striations operator – Line segments show the orientation of any texture pattern in a small window
(d) Sky region hypotheses – The entire scene was partitioned by the KNIFE segmenter. Each region that is above the geometric horizon, is relatively bright, and is relatively untextured is displayed. No regions survived.	(e) Thin object hypotheses – A linear delineation operation was performed on the output of the homogeneity operator (b) above. Short segments and highly-convoluted segments were removed. Many of the tree trunks have been identified.	(f) Ground region hypotheses – Regions of horizontal striations were extracted from (c) above. Small regions have been discarded. Very little ground was identified because the terrain is covered with tall grass.

Figure 4: Output of various operators applied to a natural scene

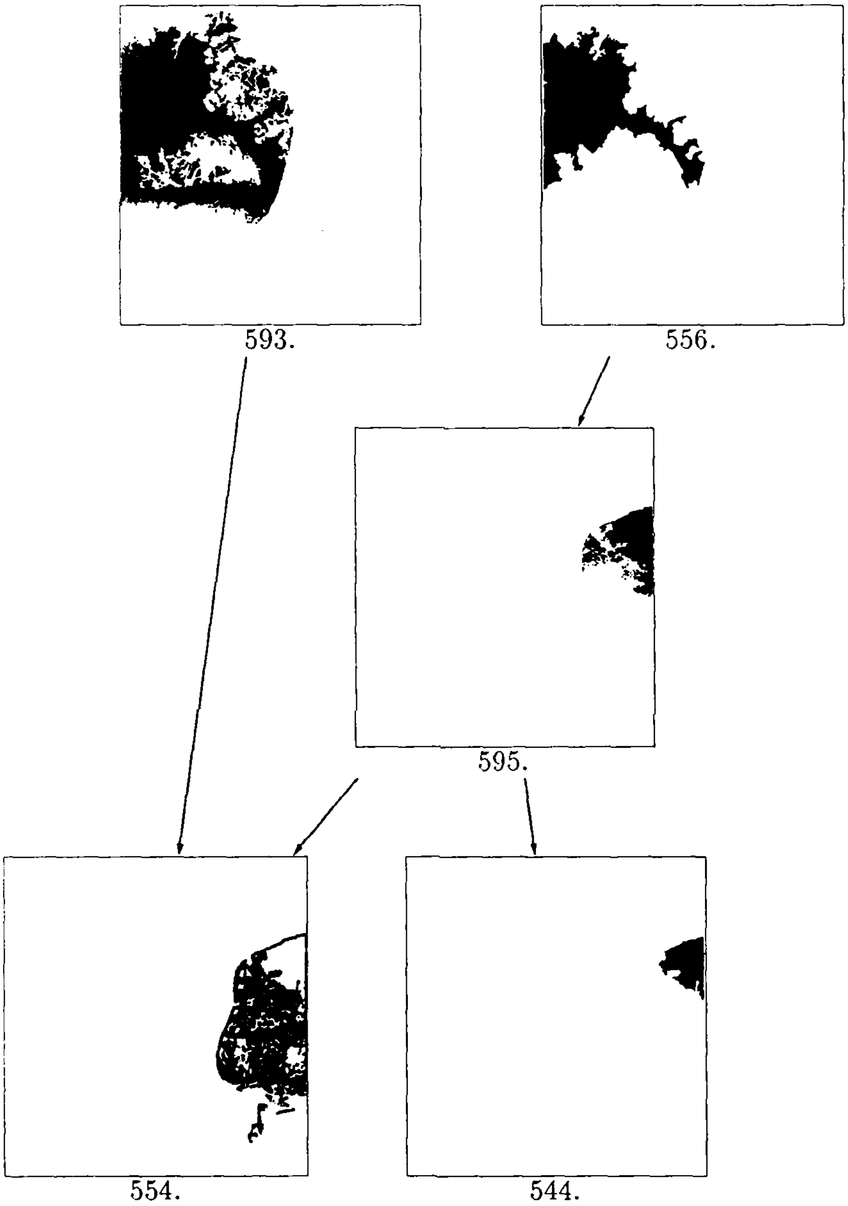


Figure 5: Partial order of sky candidates for the tree image of Figure 8

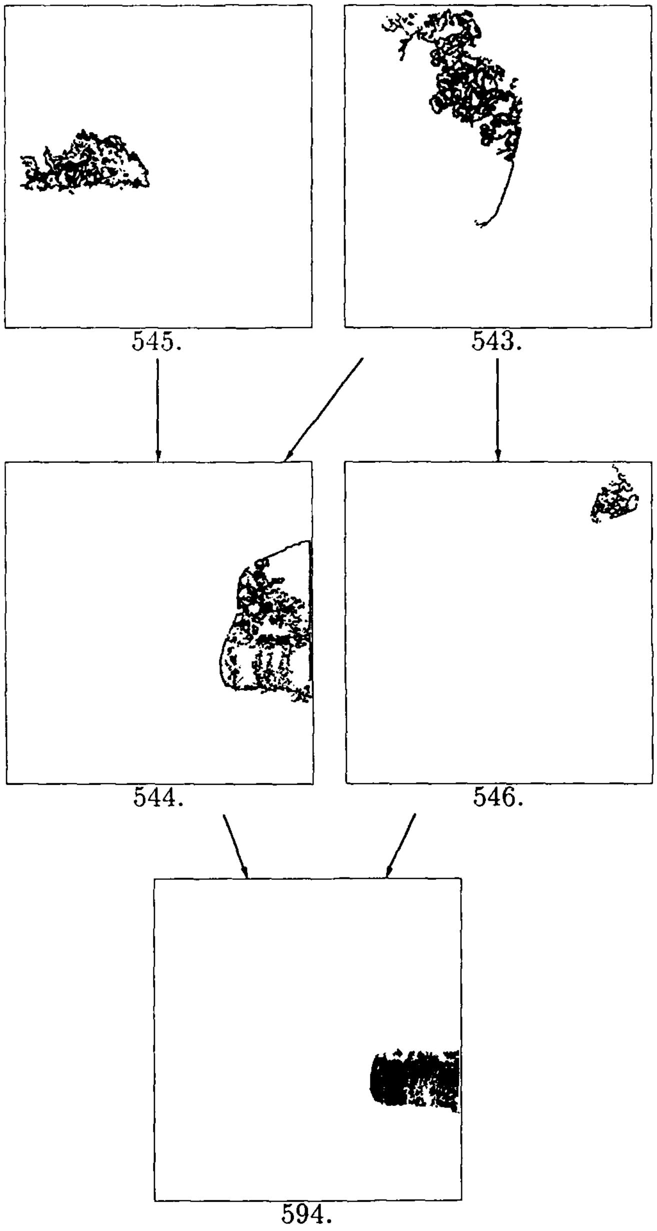
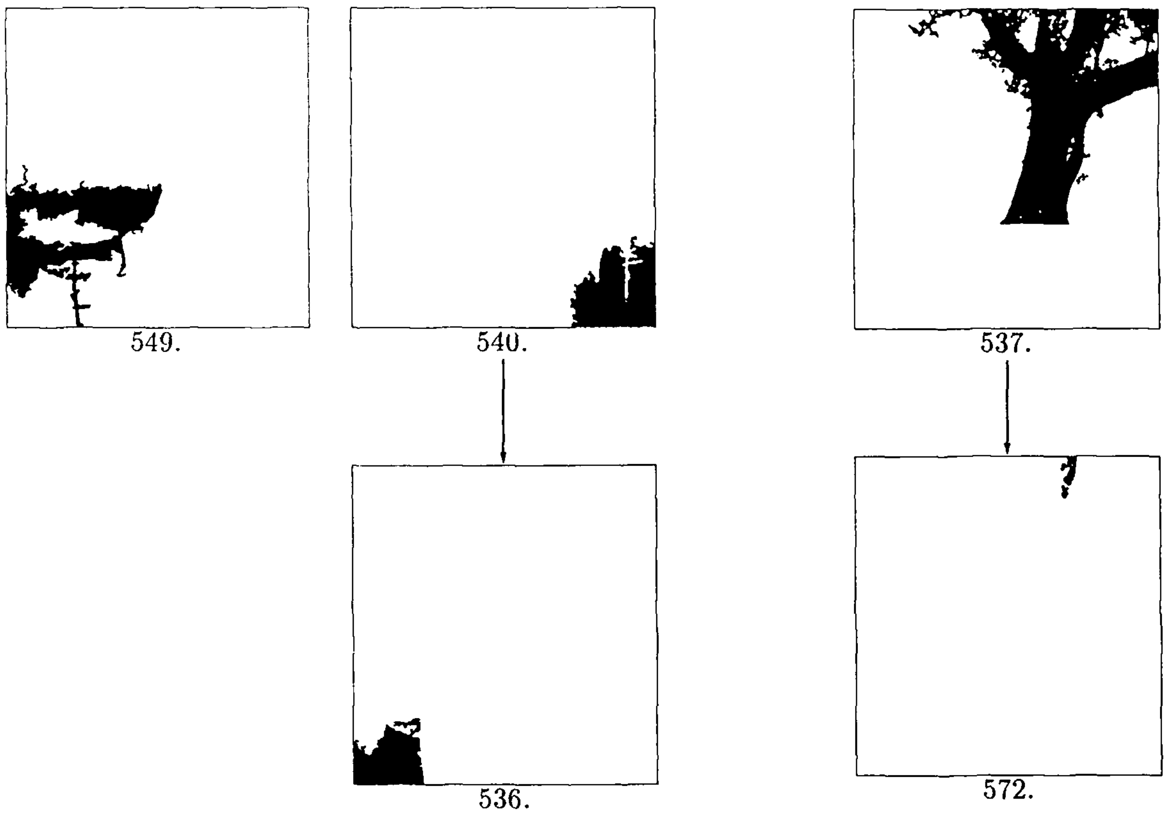


Figure 6: Partial order of foliage candidates for the tree image of Figure 8



(a) Ground candidates

(b) Tree trunk candidates

Figure 7: Partial orders for the tree image of Figure 8

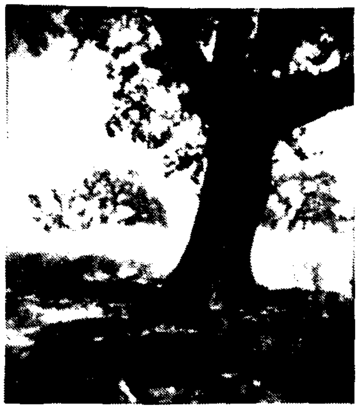
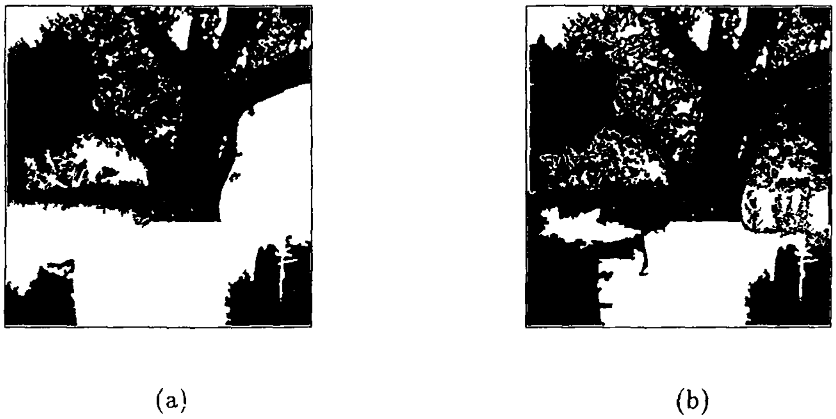


Figure 8: A natural scene of a tree on the Stanford campus



(a) (b)

Figure 9: Region coverage maps for two cliques formed from the Stanford tree scene

GeoMeter: A System for Modeling and Algebraic Manipulation

C. I. Connolly[*] D. Kapur[†] J. L. Mundy[‡]
R. Weiss[*]
Computer and Information Science Department
University of Massachusetts at Amherst [§]
Artificial Intelligence Program
GE Corporate Research and Development Center [¶]

March 30, 1989

Abstract

The *GeoMeter* modeling system is described. The system is designed to manipulate solid models for a variety of purposes. *GeoMeter* also supports polynomial and transcendental function manipulation, including methods for solving systems of polynomial equations. The applications for such methods in the context of solid modeling and computer vision are also discussed.

1 Introduction

GeoMeter [16] is a system written in Common Lisp for the purpose of modeling solid objects and providing tools for algebraic manipulation. The original motivation for *GeoMeter* was as a library to support experiments in Computer Vision, although its uses are by no means limited to that application. It is the result of several years of effort in both the Image Understanding Laboratory at the GE Research and Development Center, and more recently in the VISIONS Group at the University of Massachusetts in Amherst. Others who have been affiliated with this software are now at the Rensselaer Polytechnic Institute and the State University of New York at Albany.

Existing uses of *GeoMeter* include robot navigation [17,18], model matching [33], model construction from image data [32], proofs of geometry theorems using algebraic techniques [14], and computation of generic view information [19]. *GeoMeter* is written in Common Lisp, and has been compiled and tested on TI Explorers, Symbolics Lisp Machines, VAXLisp, and Suns under Lucid Lisp. Work is underway to allow *GeoMeter* to run on the Sequent Balance 2000 series computer.

Many of the functions and data structures in *GeoMeter* have close counterparts in mathematics. The implementors attempted to approach classical mathematical terminology in naming functions and data structures. The intent was to keep interested users from being bewildered by a deluge of nonstandard terminology. In addition, it allows users to resort to their own mathematical references for clarification of certain concepts, when desired.

1.1 Representations

There are many different representations for encoding the shape and three-dimensional structure of objects. All of them impose some type of restriction on the surfaces that can be described. For example, ACRONYM uses generalized cylinders as the basic primitive [5], SuperSketch uses superquadrics [29], some are specifically polyhedral [31] while others provide multiple primitives [6] For example, the Designer system [30] has rectangular blocks together with spheres, cylinders, and tori.

In *GeoMeter*, the language of simplicial complexes in algebraic topology [15,20] has been adopted for describing surfaces. It provides generality and an explicit representation of edges, vertices, and faces. Each of these serve as a type of geometric primitive, and can be parameterized as a smooth function from a point, unit interval, and triangle

[*] University of Massachusetts at Amherst

[†] State University of New York at Albany

[‡] GE Research and Development Center

[§] Supported in part by the following: NSF/CER Grant DCR8500332, DARPA Grant F30602-87-C-0140

[¶] Supported in part by the following: DARPA Contract DACA76-86-C-007, AFOSR Contract F49620-89-C-003

to $\mathbf{R}^3$, respectively. For example, a standard 0-simplex is a point, a 1-simplex is a straight line segment, and a 2-simplex is a triangle (see figure 1). In the usual mathematical approach, a smooth n-simplex is a differentiable map from the standard n-simplex to a subset of $\mathbf{R}^3$, and the images of these 0-, 1-, and 2-simplices correspond to the vertices, edges and faces of a surface. Surfaces are thus constructed as the union of these primitives, and are denoted by an algebraic sum of simplices. This representation produces a triangulation of the surface, where the triangles are not necessarily planar. Each smooth simplex determines an orientation on its image, i.e. a choice of the direction of the normal at each point. It is worth noting that the theory of Algebraic Topology provides operations for determining whether the triangles in a simplicial complex fit together to form a closed surface. This theory provides the foundation for many of *GeoMeter*'s operations.

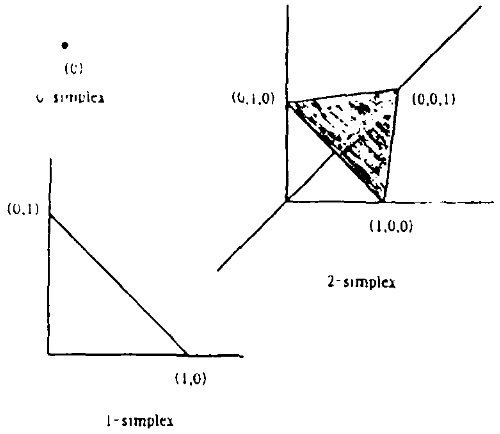


Figure 1: 0-, 1-, and 2-simplices

2 GeoMeter Structure

GeoMeter has two basic parts: a geometric section, and an analytic section. The geometric section consists of those functions and data structures which are used to describe physical objects. The analytic section consists of functions and data structures used in manipulating polynomials and transcendental functions.

2.1 Geometric section

The three basic entities which *GeoMeter* uses to represent sets are the vertex, the edge, and the face. These entities are composed to represent solid objects. The vertex is a 0-dimensional primitive which has an x, y, z position in space. An edge is a 1-dimensional set defined by two vertices (if linear). Edges can also be defined by three bounded univariate functions (if parametric) or as the planar zero set of a bivariate polynomial (if implicit). Linear edges have a direction vector and a normal vector (defined with respect to the origin). A face is a 2-dimensional set defined by a collection of edges. In *GeoMeter*, faces can also be defined parametrically and implicitly. Planar faces have a normal vector and a transformation matrix to define their coordinate system.

Interleaved with the faces, edges, and vertices are topological structures which are used to define the connectivity of sets in the model. A 0-chain is a set of vertices from which an edge can be defined. A 1-chain is a set of edges from which a face can be defined. A 2-chain is a set of faces which can be used to define a surface. An important concept in forming closed surfaces, i.e. objects, is the definition of the boundary. Every 2-chain has a boundary which is a 1-chain. If the boundary of a 2-chain is 0, then the 2-chain is said to be a 2-cycle. Similarly, if the boundary of a 1-chain or 0-chain is 0, it is a cycle. Each of *GeoMeter*'s chain structures can be used to represent cycles. A 1-cycle, i.e., a chain in which every vertex is used on exactly two edges, forms one or more polygons in the plane. [1] Likewise, a 2-cycle defines one or more polytopes.

[1] Note that this differs slightly from the usual definition.

Objects are built hierarchically starting with vertices. Vertices can be added together to form a 0-chain, a 0-chain with two points can be used to create an edge, edges can be added together to form a 1-chain, 1-chains can be used to create faces, etc. Usually, for computational simplicity, straight lines are used for edges and planes for faces, so that curved surfaces are approximated by polyhedra. Models can also be built by joining faces along common edges. A hinge function enforces the constraints that the edges must be aligned. Due to the generality of the mathematical framework, it is possible to represent semi-algebraic curves and surfaces, and there are some procedures for manipulating these objects. There are plans for the future to expand this capability. In addition, *GeoMeter* is capable of representing superquadrics and generalized cylinders.

2.2 Analytic section

A major section of *GeoMeter* is devoted to the manipulation of polynomials and transcendental functions. The motivation for these functions is twofold. They permit the exact description of curved surfaces. They also provide the mechanism for performing algebraic deduction, which is useful in reasoning about geometric relations. *GeoMeter* provides polynomial arithmetic and various ordering and testing predicates for polynomials. A full set of utilities for printing and evaluating polynomials and transcendental functions is also available.

GeoMeter allows several specialized operations on polynomials. Functions for performing polynomial arithmetic, GCD, univariate factoring, remainder sequences, decomposition, resultant computation, and Gröbner Basis [7] computation are incorporated into *GeoMeter*. Functions are available for bounding the roots of univariate polynomials and limited capabilities exist for performing Cylindrical Algebraic Decomposition [2]. Methods for triangulating sets of polynomials are available, as well as functions for testing the consistency of a polynomial with a triangulated set. Polynomials are represented in distributed form. Every polynomial is a list of monomials. In turn, each monomial is a cons pair consisting of a coefficient and a *term* representing a power product of the variables of the polynomial.

Transcendental functions are represented as a pair (p, s) where p is an arbitrary polynomial, and s is a set of substitutions mapping the variables of p to functions. For instance, a circular arc in *GeoMeter* is represented by a pair of transcendental functions:

$$\begin{aligned} p_1(x) &= rx + x_0, \quad x \rightarrow \cos\theta \\ p_2(y) &= ry + y_0, \quad y \rightarrow \sin\theta \end{aligned}$$

where x_0, y_0 is the center of the arc and r is its radius.

3 Applications

3.1 Representing curves and surfaces

As mentioned above, *GeoMeter* has the capability to define parametric surfaces and curves. Both are defined using transcendental or polynomial functions in one or two variables over some interval. Parametric curves are defined using three transcendental functions in one variable, u: $x(u), y(u), z(u)$. The curve structure also has a slot for the bounds on u and the sampling rate for displaying the curve. Parametric faces are defined similarly, but the defining functions and interval are bivariate.

GeoMeter also supports algebraic curves and faces. Algebraic Faces are implicit surfaces defined by a polynomial $p(x, y, z) = 0$. In conjunction with algebraic faces, *GeoMeter* can also represent planar algebraic curves expressed with bivariate polynomials. Decomposition techniques [1,2] are used for display and analysis of such curves and surfaces. Figure 2 shows a sample *GeoMeter* frame displaying various objects.

3.2 Projection and Image Formation

There are a number of applications which require models of the imaging process. Modeling the projection process is not only useful for display of objects. It has been used (via *GeoMeter*) to model appearances for robot navigation [18,17]. The projection process within *GeoMeter* is central projection, also known as perspective projection. The projection is modeled in *GeoMeter* by a Camera entity, which contains the projection parameters. The projection is computed on points in $\mathbf{R}^3$, which are represented by homogeneous coordinates in $\mathbf{R}^4$. Each point is rotated and translated by a homogeneous 4x4 matrix that represents the transform from model coordinates to camera coordinates. Then each point is projected onto the image plane according to the camera parameters: the camera lens focal length, zoom, aspect ratio, and the image center.

In implementing the projection operation, we are not only interested in point sets, but also in the edges and possibly the faces that constitute the model as well as their visibility. *GeoMeter* contains a Viewer entity, which stores information about what is to be projected and the camera entity parameters for performing the projection.

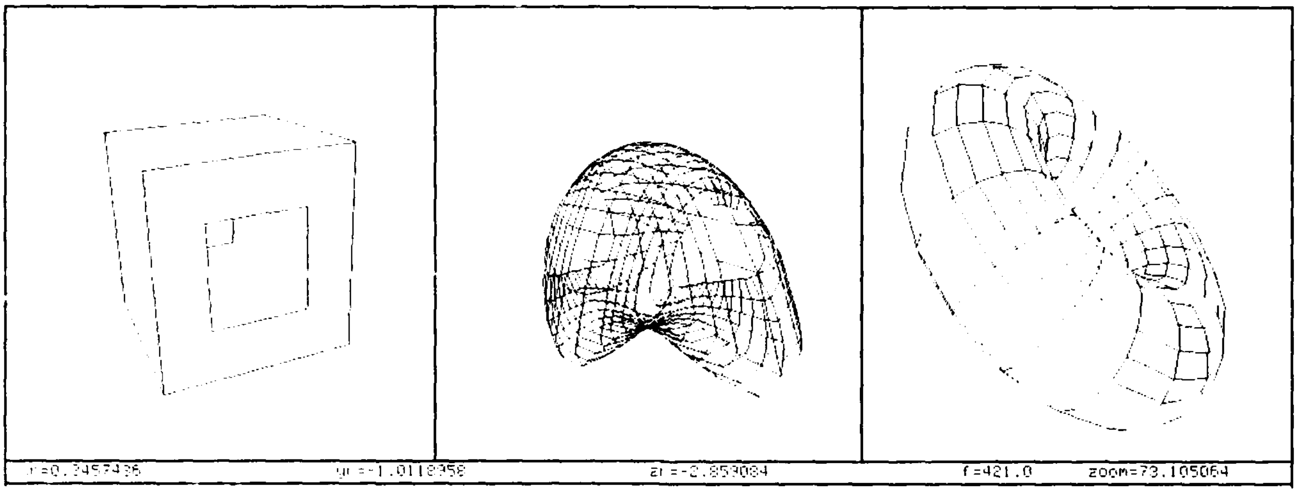


Figure 2: A Planar, algebraic, and parametric surface, respectively

In addition, the Viewer contains the global, local, and projected coordinates of the points of the model, along with incidence information that allows faces and edges to be selected and drawn. The global coordinates and the incidence information are obtained from the faces, edges, and vertices of the model (or part thereof) that is being projected.

3.3 Model construction

GeoMeter has been used for experiments in model construction at GE. Stenstrom, et al. [32] describe a method for constructing volume models from image data. The technique involves the formation of volume sets from different views and performing boolean intersection of the sets obtained. Most of the implementation of this technique is carried out in *GeoMeter*. Other techniques have been developed [3] which use algebraic constraints to construct "parameterized models". Extensions of this work are reported later in section 4.3.2, and elsewhere in these proceedings [27].

3.4 Robot Navigation

GeoMeter is also being used for robot navigation experiments at the University of Massachusetts [17,18]. In order to meet the demands of a robot navigation system, *GeoMeter* had to satisfy several requirements. Geometric modeling must be easily interfaced with other modules needed for specific tasks in navigation. In our work it is interfaced directly with the high-level model matching functions used during model-to-image and image-to-model matching [4]. Components of the models are annotated with visual characteristics. A multilevel representation scheme is required so that parts of objects can be isolated and named as landmarks for recognition.

For navigation through a complex domain, one needs to model the world in which objects can be located. In our environment, buildings, lamp posts, and telephone poles must be modeled. Buildings have sub-objects such as windows and doors. Elements at all levels may be annotated with information relevant to visual tasks. In order to navigate from known landmarks, the environment must be modeled accurately. For the University of Massachusetts campus, this was done using information obtained from a careful survey of the environment, building plans, and direct measurements. Once the landmarks are identified, *GeoMeter* is used for pose refinement [25] to obtain robot bearings from visual information and information provided by the campus model.

3.5 Matching

Both at GE and at the University of Massachusetts, experiments in object recognition are underway [8,33]. These experiments use *GeoMeter* for some of the geometric operations and data structures required for correspondence and the computation of transformations. In both schemes, models are created, stored and displayed using *GeoMeter*. Images are processed to obtain edge information which is then compared to the model data to identify objects and their poses. In work described elsewhere in these proceedings, a method has also been devised for selecting and verifying the best matches out of a finite set of possibilities [21].

4 Solution Techniques

Solution of nonlinear systems of equations and optimization are two functions which are central to some of the aforementioned application areas. *GeoMeter* supports methods for solving such problems. Much of this machinery is oriented toward exact solution methods such as Wu's Method [34], the Gröbner Basis method [7,22] or algebraic decomposition [2,9]. There are also functions for computing approximate solutions.

4.1 Numerical methods

GeoMeter uses numerical methods for obtaining approximate solutions to nonlinear systems of equations. Functions exist for exact computation of the Jacobian, and for Newton's method for nonlinear systems. Morgan [26] has recently developed a continuation method for solving algebraic systems numerically. This is a promising method that avoids many of the convergence problems to which Newton's method is susceptible. Work is being initiated to incorporate this method into *GeoMeter*. In addition, most of the computation used for modeling purposes in *GeoMeter* is numerical in nature (e.g., curve and surface intersection, boolean operations, transformation, etc.).

4.2 Support for Geometric Reasoning

GeoMeter supports two different but related approaches to reasoning in algebraic geometry: a refutational method based on the Gröbner basis algorithm [23] and a direct method by Wu based on the Ritt principle. [2] Both methods take polynomial equations as input. It is assumed that geometric relations have already been transformed into polynomial equations.

4.2.1 Refutational approach based on the Gröbner basis method

In the refutational method, the hypotheses of a geometry statement and the negation of the conjecture being proved are input and it is checked using the Gröbner basis algorithm that they are not satisfiable. If the algorithm detects that the Gröbner basis includes 1, it declares that the conjecture follows from the input. Otherwise, the Gröbner basis generated by *GeoMeter* can be used to extract out additional conditions that must be imposed on the input for the conjecture to follow from the input.

The refutational method has been shown to be complete for deciding whether a conjecture follows from the input or not [23]. In the case when the conjecture does not follow from the input, the method has also been shown to be complete for computing conditions under which the conjecture would follow from the input. The method has been successfully used to prove over a hundred geometry theorems including many nontrivial theorems which even humans find very difficult to prove. The method is fully described with examples and theoretical foundations in [23].

4.2.2 The direct approach based on Wu's method

GeoMeter also supports Wu's method for geometric reasoning [34,36]. In contrast to the refutational approach based on the Gröbner basis algorithm, the method is direct. The hypotheses of a geometry statement are transformed into a triangular form using the Wu-Ritt method. *GeoMeter* expects the user to specify the independent variables as well as a total order on dependent variables; it currently does not provide any assistance in selecting dependent variables. Independent variables correspond to the degree of freedom in a geometric configuration defined by a geometry statement. Intuitively, independent variables are those variables which can be assigned arbitrary values and which determine the values of dependent variables.

Once a triangular system of polynomials is computed, the polynomial corresponding to the conjecture is pseudo-divided by each of the polynomial equations in the triangular form to successively eliminate each dependent variable in the conjecture. If the remainder is 0, then conjecture follows from the hypotheses. In this case, the method also identifies subsidiary conditions ruling out degenerate cases for the conjecture to follow from the hypotheses.

If the remainder is not 0, then it is still possible that the conjecture follows from the hypotheses. The triangular form of the hypotheses must be checked for irreducibility. If polynomials in the triangular form cannot be factored over successive extension fields, then the triangular form is irreducible. If the remainder of a conjecture with respect

[2] In fact, this portion of the software used to be called GEOMETER [13] and the whole system used to be called GEOCALC until we discovered that there was a commerical product with the name GEOCALC. It was then decided to call the whole system *GeoMeter*.

to an irreducible triangular form obtained from the hypotheses is not 0, then the conjecture does not follow from the hypotheses. Otherwise, the polynomials in the triangular form must be factored generating a set of irreducible triangular forms; the conjecture must then be checked over each of these irreducible triangular forms. *GeoMeter* does not provide algorithms for checking the irreducibility of a triangular form, nor does it provide any algorithms for factoring over extension fields. Theoretical foundations of Wu's method are discussed in [35]. An excellent implementation of Wu's method and its success in proving nontrivial geometry theorems are discussed in [10,11]. An informal discussion of Wu's method and its application to problems in perspective viewing is described in [24].

4.3 Hybrid Approaches

4.3.1 Hybrid Solution Methods

The power of purely exact methods for geometric reasoning and representation is limited to relatively small problems (see [9] for an analysis). By contrast, large model specifications consisting of thousands of entities can be successfully processed quickly if numerical methods are used. While they are capable of fast solution of such systems, numerical methods can be plagued with error accumulation. More importantly, Newton's method is only guaranteed to converge under very strict conditions [28].

Using tools in *GeoMeter*, an approach is being pursued where exact methods are used to improve the convergence properties of numerical methods. The basic idea is to determine a set of independent model parameters which can be freely varied with respect to the model constraints. Exact methods are then used to triangulate the constraint equations, as in Wu's method described earlier. The triangulated constraints can then be easily differentiated to determine the singularities of the Jacobian matrix. Thus, algebraic techniques can be used to implement restrained versions of Newton's method which only operate in "safe" regions where gradients are always uniquely defined. Experiments with this basic technique have been successful on systems with up to eight parameters. Other experiments are underway to examine the possibility of reduction and decomposition of the original system. This involves decomposing the system into individual subproblems which can be solved more easily than the original problem.

4.3.2 Constraint-based modeling

One approach to constraint-based modeling is to represent geometric constraints in a relational database, along with a numerical specification [12]. Known relationships can be retrieved using standard database techniques. Additional relationships can be derived by computation on the numerical specification of the objects and object relationships, or by logical inferences on the known relations. The power of automated logical inference techniques is limited to relatively simple deductions. On the other hand, the inference of relationships by numerical processes alone has limited robustness, particularly in the case of empirical data with significant errors. The numerically derived relationships can be easily inconsistent with logically derived relations. SRI's CKS (Core Knowledge System) deals with this uncertainty by providing a logic of belief which can handle multiple agents with various levels of reliability.

Another approach is to maintain a consistent set of geometric relationships that are maintained in a relational network, but specified algebraically [30]. The algebraic equations and inequalities provide a parametric specification of the objects and object relationships. The interaction with empirical data is taken as a problem in error minimization. That is, the parameters of the model specification are to be adjusted such that the distance between model predictions and actual image features, and other empirical data, is a minimum. The resulting model configuration maintains the consistency of *a priori* constraints while accommodating empirical relationships as closely as possible. We refer to this approach as constraint-based modeling [27].

The next major development of the constraint-based modeling technique in *GeoMeter* will be the integration of the algebraically derived convergence strategy with classical nonlinear programming methods.

5 Conclusion

GeoMeter is a versatile tool for solid modeling and algebraic manipulation. It is publicly available via anonymous FTP from Internet host VAX1.CS.UMASS.EDU (128.119.40.1), from the directory

```
VIS$DISK:[GEOMETER]*.LISP
```

Documentation is available in LaTeX form from the directory

```
VIS$DISK:[GEOMETER.DOC]
```

The authors would like to acknowledge the many people who have contributed to *GeoMeter*, especially Michele Barry, Brian Burns, Dave Cyrluk, Antonio Delgado, Rick Harris, Aaron Heller, Tim Kelliher, Ross Stenstrom, Dan Thompson, and Patricia Vrobel.

References

[1] Dennis S. Arnon. Topologically reliable display of algebraic curves. *Computer Graphics*, 17(3):219–227, July 1983.

[2] Dennis S. Arnon, George E. Collins, and Scott McCallum. Cylindrical algebraic decomposition. *SIAM Journal of Computing*, 13:865–877,878–889, 1984.

[3] Michele Barry, David Cyrluk, Deepak Kapur, Joseph Mundy, and Van-Duc Nguyen. A multi-level geometric reasoning system for vision. *Artificial Intelligence*, 37:291–332, 1988.

[4] J. Ross Beveridge, R. Weiss, and E. Riseman. Matching and fitting sets of 2d line segments to broken and skewed data. Technical Report in preparation, COINS department, University of Massachusetts at Amherst, 1989.

[5] Rodney A. Brooks. *Symbolic Reasoning Among 3-D Models and 2-D Images.* PhD thesis, Stanford University, June 1981.

[6] C. Brown. PADL2: A Technical Summary. *Proc. IEEE Computer Graphics Applications*, 2(2):69–84, March 1982.

[7] Bruno Buchberger. *An Algorithm for Finding a Basis for the Residue Class Ring of a Zero-Dimensional Polynomial Ideal.* PhD thesis, Universität Innsbruck, Innsbruck, Austria, 1965. in German.

[8] J. Brian Burns and Leslie J. Kitchen. Rapid object recognition from a large model base using prediction hierarchies. In *Proceedings: Image Understanding Workshop*, pages 711–719. DARPA, Morgan-Kaufman, Inc., 1988.

[9] John Francis Canny. *The Complexity of Robot Motion Planning.* PhD thesis, Massachusetts Institute of Technology, 1987.

[10] S.-C. Chou. Proving elementary geometry theorems using Wu's algorithm. In W. W. Bledsoe and D. W. Loveland, editors, *Contemporary Mathematics*, volume 29, pages 235–241. 1984.

[11] S.-C. Chou. *Proving and discovering theorems in elementary geometry using Wu's method.* PhD thesis, University of Texas, Austin, TX, 1985.

[12] N.R. Corby, J.L. Mundy, P.A. Vrobel, A.J. Hanson, L.H. Quam, G.B. Smith, and T.M. Strat. PACE: An environment for intelligence analysis. In *Proceedings: Image Understanding Workshop.* DARPA, Morgan-Kaufman, Inc., 1988.

[13] D. Cyrluk, R. Harris, and D. Kapur. GEOMETER: A theorem prover for algebraic geometry. In *Proceedings of the 9th International Conference on Automated Deduction (CADE-9)*, Argonne, IL, May 1988.

[14] David A. Cyrluk, Deepak Kapur, and Joseph L. Mundy. Algebraic reasoning in view consistency and parameterized model matching problems. In *Proceedings: Image Understanding Workshop*, pages 731–739. DARPA, Morgan-Kaufman, Inc., April 1988.

[15] S. Eilenberg and N. Steenrod. *Foundations of Algebraic Topology.* Princeton University Press, Princeton, NJ, 1952.

[16] C. Connolly et. al. *GeoMeter: Solid Modelling and Algebraic Manipulation.* COINS Department, University of Massachusetts at Amherst, 1989.

[17] Claude Fennema, Allen Hanson, and Edward Riseman. Towards autonomous mobile robot navigation. In *Proceedings: Image Understanding Workshop.* DARPA, Morgan-Kaufman, Inc., 1989.

[18] Claude Fennema, Edward Riseman, and Allen Hanson. Planning with perceptual milestones to control uncertainty in robot navigation. In *Proc. of SPIE*, Cambridge, MA, November 1988. International Society for Photographic and Industrial Engineering.

[19] Peter Giblin and Richard Weiss. Reconstruction of surfaces from profiles. In *Proceedings: Image Understanding Workshop*, pages 900–908. DARPA, Morgan-Kaufman, Inc., February 1987.

[20] M. Greenberg and J. Harper. *Algebraic Topology A First Course.* Benjamin, Reading, MA, 1981.

[21] A. J. Heller and J. R. Stenstrom. Verification of recognition and alignment hypotheses by means of edge verification statistics. In *Proceedings: Image Understanding Workshop*. DARPA, Morgan-Kaufman, Inc., 1989.

[22] A. Kandri-Rody and D. Kapur. Algorithms for computing the Gröbner bases of polynomial ideals over various Euclidean rings. In *Lecture Notes in Computer Science*, volume 174. Springer-Verlrg, New York, NY, 1984.

[23] D. Kapur. A refutational approach to geometry theorem proving. *Artificial Intelligence*, 37:61–93, 1988.

[24] Deepak Kapur and Joseph L. Mundy. Wu's method and its application to perspective viewing. *Artificial Intelligence*, 37:15–36, 1988.

[25] Rakesh Kumar. Determination of camera location and orientation. In *Proceedings: Image Understanding Workshop*. DARPA, Morgan-Kaufman, Inc., 1989. also COINS Technical Report (in preparation), University of Massachusetts at Amherst.

[26] Alexander Morgan. *Solving Polynomial Systems Using Continuation for Engineering and Scientific Problems*. Prentice-Hall, Inc., Englewood Cliffs, NJ, 1987.

[27] J. L. Mundy, P. A. Vrobel, and R. E. Joynson. Constraint-based modeling. In *Proceedings: Image Understanding Workshop*. DARPA, Morgan-Kaufman, Inc., 1989.

[28] J. M. Ortega. *Numerical analysis - A second course*. Academic Press, New York, NY, 1972.

[29] Alex Pentland. Perceptual organization and the representation of natural form. *Artificial Intelligence*, 28:293–331, 1986.

[30] Robin Popplestone. The Edinburgh Designer System as a Framework for Robotics: The Design of Behavior. *AI EDAM*, 1(1), 1988.

[31] A. Requicha and H. B. Voelker. Solid modeling: A historical summary and contemporary assessment. *IEEE Computer Graphics and Applications*, 2(2):9–24, March 1982.

[32] J. R. Stenstrom and C. I. Connolly. Model generation from images. In V. Cantoni, V. DiGesu, and S. Levialdi, editors, *Image Analysis and Processing II*, pages 269–276. Plenum Publishing Corporation, 1988.

[33] Daniel W. Thompson and Joseph L. Mundy. Three dimensional model matching from an unconstrained viewpoint. Computer Science Branch Internal Report, 1986.

[34] Wu Wen-tsün. On decision problem and mechanization of theorem proving in elementary geometry. *Scientia Sinica*, 21:159–172, 1978. also in Bledsoe and Loveland, eds., *Theorem Proving: After 25 Years, Contemporary Mathematics.*

[35] Wu Wen-tsün. Basic principles of mechanical theorem proving in geometries. *J. Syst. Sci. Math. Sci.*, 4(3):207–235, 1984.

[36] Wu Wen-tsün. Some recent advances in mechanical theorem proving of geometries. In W. W. Bledsoe and D. W. Loveland, editors, *Contemporary Mathematics*, volume 29, pages 235–241. 1984.

Adaptive Model Base Indexing

Thomas M. Breuel

Center for Biological Information Processing
and
MIT Artificial Intelligence Laboratory
Cambridge, MA 02139, USA

Abstract

We present a new approach to the model base indexing stage of visual object recognition. The key ideas in our approach are that we use the same processing and feature extraction steps during the learning (model acquisition) and execution phase of visual object recognition, and that we encode the relative spatial locations of features in a view compactly as a bit vector, the "combined feature vector".

As a consequence of these measures, model acquisition algorithms based on the combined feature vector have a number of desirable properties: they do not make specific assumptions about the analytic relationship between the viewing parameters and feature locations in the image; they can change the degree of quantization (and thereby the degree of generalisation to new viewpoint and new shapes) adaptively; they represent object models as collections of compactly encoded object views ("object signatures"), and therefore can represent partial object models easily; and they make the statistical structure of the model acquisition and object recognition problem more explicit and thereby accessible to standard statistical techniques.

Empirical results from applying our algorithm to the limited domain of wireframe polyhedral objects are promising. In addition, we believe that it has relevance to psychological, psychophysical and neurobiological theories of object recognition.

1 Introduction

Feature based visual object recognition is based on the following observation. Assume we know the shape of an object and the positions of several points on the object. If we can identify at least three such points (features) in an image, we can recover the parameters of the viewing transformation. If we can identify more than three features, the equation for recovering the viewing transformation is overconstrained, and it is very likely that the only object model that will allow an (approximately) consistent solution of the equation for the viewing transformation is the object model corresponding to the object in the image.

Feature based approaches to visual object recognition have met with some success (without claiming completeness, some recent relevant work on the subject can be found in Shirai, 1981, Grimson, 1984, Baird, 1985, Grimson and Lozano-Perez, 1985, Huttenlocher and Ullman, 1987, Goldberg and Lowe, 1987, Lowe, 1987, Grimson, 1988, Cass, 1988).

However, they have several undesirable properties:

1. They assume that features are rigidly attached to the surface of an object; they assume that the viewing transformation takes a fixed, known parametric form. There are many kinds of features for which this assumption does not hold, but which are nevertheless very useful for identifying objects (see Figure 1). They require labels to be assigned to features consistently across different views. They get rather complex if objects are parameterized.

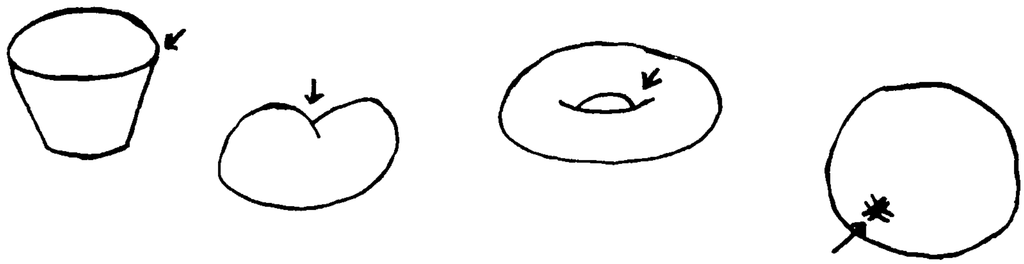


Figure 1: Examples of objects and features where features that are easily detected in images or even line drawings are not fixed to the surface of the object.

2. They assume that features on the object and features in the image can be put into one-to-one correspondence (labelled uniquely), or they require expensive search techniques to try out many such correspondences.

3. They require 3D object models, or, at least, require that the relative 3D positions of the features on the object are known. Many feature based approaches are therefore difficult to apply when object models must be acquired automatically from images with sparse or no depth information.

4. They often use rather arbitrary metrics to determine whether the locations of the features in the image are close enough to the locations predicted from the model and the viewing transformation. But the degree to which a particular arrangement of features suggests the presence of a particular object in the image depends on many factors, some of which are not even geometric in nature (see Figure 2 for some examples).

In this paper, we describe an approach to feature based visual object recognition that addresses these problems. We are primarily concerned with the model base indexing part of recognition, i.e. the identification of candidate object models that may be present in a given image; these candidates are then verified using other methods. Also, what we will refer to as an "object" may be considered an "object part" by some.

2 The Method

We assume that the low level vision modules compute a collection of feature types and feature locations that are used as input to our recognition (indexing) stage. By a "feature" we mean any localizable property of the $2\frac{1}{2}$D sketch (Marr, 1982), for example midpoints of line segments, intersections of line segments (vertices, T-junctions, intersections of the extensions of three or more line segments, *etc.*), local maxima of intensity (often corresponding to specular reflections), local depth maxima (viewer centered), centers of gravity of regions and "blobs", intersections of occlusion boundaries, maxima of curvature on occlusion boundaries (see Hoffman and Richards, 1983, for a discussion of the importance of such features), and possibly many others. Some of these features are fully characterized by their type and their position in

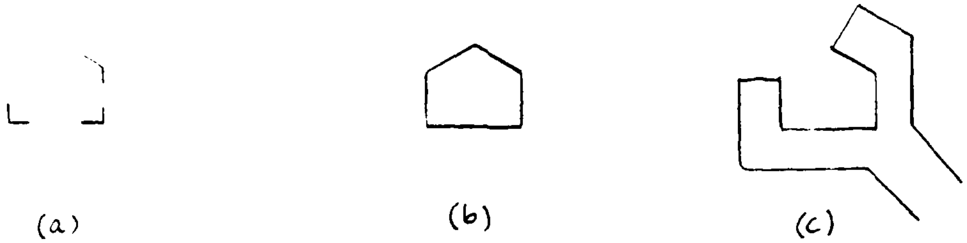


Figure 2: How well a particular arrangement of features in the image allows us to identify a particular object can depend on non-geometric factors. The features shown in (a) are very characteristic of object (b); however, if object (c) is also present in the model base, we need additional evidence to distinguish the two objects.

the image, while others have additional properties associated with them, such as orientation in the case of line segments, or second order moments in the case of blobs.

Previous approaches to feature based object recognition assume that we have object models that allow us to predict the locations and properties of the features in the image given the viewing transformation. To find analytically an object model and viewing parameters consistent with the feature positions in a given image, we have to decide on which image features correspond to which object features first, and we can then try to find a solution to the equation relating viewing parameters, image features, and object features (the "viewing equation").

If (at least some) features are distinguishal :- by "labels" or "types", we can use these to establish an initial correspondence between image and object features (Huttenlocher and Ullman, 1987). If no such information is available, we can use a search strategy (Grimson, 1984, Grimson and Lozano-Perez, 1985, Grimson, 1988). A more recent approach that is implicit in our method and has been proposed independently by Lamdan and Wolfson, 1988, is Geometric Hashing. Rather than trying to find a solution to the viewing equation explicitly, the locations of features in the image plane are precomputed for each object and for a significant number of different views and encoded in a hash table. When an object is to be identified in an image, the precomputed views corresponding to the object models are matched against the image using hashing and voting techniques. Geometric Hashing avoids some of the complexity problems of the analytic techniques, but because of its reliance on explicit object models still has many of the disadvantages mentioned in the introduction.

In order to avoid these problems, and in order to obtain a system that can acquire object models incrementally, we proceed as follows. During the *learning phase*, the system is presented with real (or realistic, synthesized images) from a variety of different viewpoints. Features are extracted from the training images using the same mechanisms used later for the test images. Then the relative positions and orientations of the features in the image are encoded (algorithm **ENCODE**) into a "combined feature vector" using hashing and quantization techniques. All the feature vectors obtained during the learning phase are stored and combined into a data base (algorithm **STORE**). During the *execution phase*, test images are encoded using the same algorithm **ENCODE** that was used to build the data base, the resulting combined feature vector is matched (algorithm **MATCH**) against the images in the data base, and the best match is

1. Input: Some collection of features in the image plane

2. Compute a description of the features that is invariant under 2D translation, rotation, scale

   angles-1: <27,39,45> relative-angle: 14 angles-2: <33,21>

3. Quantize (round) the description

   angles-1-qt: <20,40,40> relative-angle-qt: 20 angles-2-qt: <40,20>

4. Convert the description into a number (a one-to-one function), giving the unhashed combined feature

   18253100

5. Hash the number from the previous step down into a manageable range, giving the (hashed) combined feature

   701

6. Set the bit indexed by the combined feature in the combined feature vector

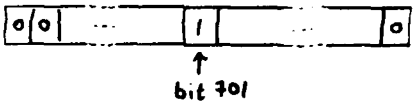


Figure 3: Algorithm **ENCODE**: Encoding collections of image features into the combined feature vector.

returned.

By using the same processing steps to obtain the combined feature vector during the learning phase and the execution phase, we avoid having to know or derive the explicit relation between object shape and feature positions in the image. Furthermore, while we are building the model base, we obtain information about the statistical properties of the combined feature vector; we can therefore use statistical techniques to decide which dimensions of the combined feature vector are particularly significant for particular recognition tasks (see also Figure 2; for a discussion of "Evidential Reasoning" in visual recognition, see Lowe, 1986). And we can check on-line during the learning phase how well our data base performs and we can dynamically modify the hashing and quantization parameters in algorithm **ENCODE** accordingly.

Algorithm **ENCODE** is illustrated in Figure 3 for the case of encoding vertices into a combined feature vector. There are several important properties this encoding has:

- The encoding is invariant under 2D translation, rotation, and scaling; since there are six parameters describing the viewing transformation (assuming orthogonal or nearly orthogonal projection), this leaves two parameters that need to be covered by storing individual encoded views of an object in the data base. As a consequence, we expect the number of combined feature vectors per object to grow quadratically in the quantization used in **ENCODE**.
- The amount by which the viewpoint may change before the combined feature vector changes depends on the degree of quantization; the coarser the quantization, the larger is the area of the viewing sphere that a particular combined feature vector corresponds to. The same relation holds for variations in object shape: the coarser the quantization, the more the object shape may vary before

1. Input: a combined feature vector from the training image, a label identifying the object in the training image
2. Try to find the combined feature vector for the training example in the data base
3. If it is not present, add it with the given label
4. If it is present and has the given label, do nothing
5. If it is present and has a different label, repeat the same process using a finer level of quantization

Figure 4: Algorithm STORE: A simple algorithm to build a model base from training examples.

the combined feature vector changes. In different words, both the degree of generalization to novel viewpoints and the degree of generalization to novel shapes are controlled by the quantization. This means that as we make the quantization finer to distinguish more similar shapes, we need to obtain more training examples and store more combined feature vectors to cover the viewing sphere.

- The encoding is not invertible; a given combined feature vector could correspond to many different kinds of views of different objects (much of this is, of course, a consequence of the non-invertibility of the imaging process). However, during the computation of the combined feature vector, we can keep track of which combinations of features gave rise to which bits in the combined feature vector. This information can be useful for a later model verification stage.
- The number of *elementary features that go into* a combined feature is often so large that it overconstrains the six parameters that specify the viewing transformation.

A simple algorithm (STORE) for building a model base is shown in Figure 4. The algorithm assumes that training examples consist of isolated objects together with a label giving the identity of the object in the scene. The training image is encoded, and the encoded image is added to the data base, unless it is redundant, or unless it conflicts with a previously stored encoded view; in the latter case, the process is repeated with finer quantization in the ENCODE step.

We have not specified so far how to match (MATCH) the combined feature vectors from a novel scene against the combined feature vectors in the data base. Assume that an image consists of a number of features that belong to an object in the data base and a number of spurious features. When we combine features exclusively from the object, we get combined features that occur in the combined feature vector for the object in the image. When we combine any spurious features with any other features, we get combined features that are most likely not characteristic of any object.

The reasons why a combination of features that includes some spurious features may give rise to a valid combined feature are that the arrangement may by chance be geometrically similar to an existing valid combined feature, or because they get hashed to the save value during the hashing step in algorithm ENCODE. The former problem is a purely geometrical problem, and any purely feature based algorithm has to deal with the possibility that by chance a number of elementary features conspire to give the appearance of an object part; we will analyze the latter problem more carefully later.

From the preceeding discussion, we can see that a good choice of a distance measure for how well a combined feature vector of an image matches a combined feature in the data base is an asymmetric one,

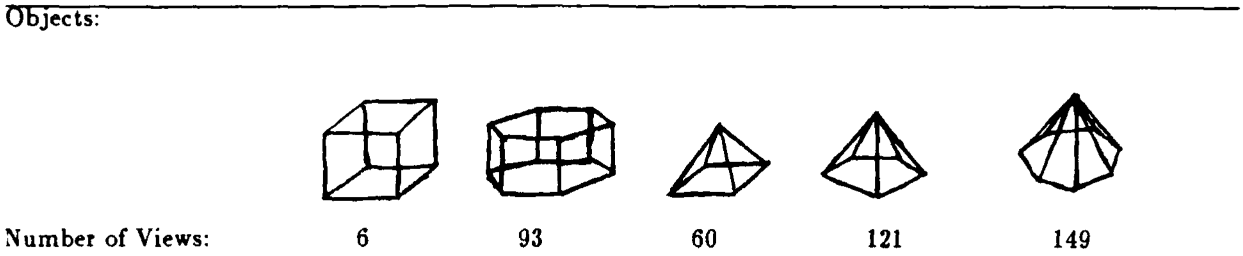


Figure 5: The number of distinct combined feature vectors (distinct "views") for the given objects. These numbers were obtained by random sampling of the viewing sphere. The degree of quantization used is sufficient to allow polyhedral objects of this kind to be distinguished with certainty. The size of the combined feature vector (the hash function) was chosen such that less than about 10% of the bits in the combined feature vectors were set.

obtained by counting the number of bits in the model combined feature vector that are also present in the image, thereby disregarding all the feature combinations that arise from context (i.e. that involve spurious features). In other words, the features originating on the object are combined to form a "signature" in the combined feature vector, and we are looking for the presence or absence of this signature, ignoring any bits contributed by context.

3 Theoretical and Empirical Results

3.1 Sample Complexity and Model Base Size

Since we store a separate combined feature vector for each "view" of an object, one might fear that the number of training examples that needs to be obtained (the sample complexity) and the number of views that needs to be stored per object is impractically large. To address this question, we have carried out some numerical experiments.

We used randomly chosen orthogonal projections of wire frame polyhedra, extracted features (vertices) and encoded the features using algorithm **ENCODE** at various levels of quantization. For the resulting combined feature vectors, we checked whether the degree of quantization was sufficient to distinguish objects reliably and counted the number of distinct combined feature vectors; some representative results are shown in Figure 5.

These are worst-case numbers, since two combined feature vectors were considered distinct even if they differed in only one bit. For accurate recognition, it is sufficient that the collection of views stored in the data base classify all new images correctly, i.e. that the combined feature vector for any new view of an object differs less from one of the corresponding training examples than from any other combined feature vector in the data base.

Our empirical results certainly do not prove that the sample complexity and storage requirements of a visual object recognition system for realistic images based on object views encoded as combined feature vectors is small; however, it does suggest that the approach might be feasible.

3.2 Robustness when Features are Missing or Spurious Features are Present

We cannot rely on the early vision modules to detect all features in the image reliably or not to add some spurious features to the image. Nor can we expect that grouping and saliency mechanisms will ensure that only features belonging to one object are used as input for our model base indexing algorithm. Empirically we find that these problems are manageable. However, we might like to obtain some theoretical worst-case bounds.

As we have already discussed, there are two kinds of possible errors; collections of features originating not from a single object could accidentally be arranged like some combined feature for an actual object, and different combined features could be hashed to the same bit in the combined feature vector. We can avoid the latter problem by making the combined feature vector large enough.

To determine how large we have to make the combined feature vector in order to keep the probability of misidentifying an object low, consider the following simple analysis. Let

- N be the number of models in the data base. We assume that the feature vectors corresponding to the models are statistically independent.
- L be the number of bits in a feature vector.
- r be the number of features in an unoccluded view of an object without any spurious features, i.e. a view corresponding to a model feature vector in the data base (we assume for the sake of simplicity that r is the same for all model feature vectors in the data base).
- n be the number of object-derived features present in the image to be classified (i.e. $r - n$ features are absent, perhaps because they are occluded or because the image is noisy).
- m be the number of spurious features in the image to be classified. We assume that the location and type of the spurious features is independent of the object present in the image.

Each feature has some unhashed feature number, say f_i^0, associated with it, and the same holds for the spatial relationship between two features[1], say that feature number is f_{ij}^1. The combined feature vector v is defined in terms of a hash function $\mathrm{HASH}(i, j, k)$ such that

$$v_l = \begin{cases} 1 & l \in \{\mathrm{HASH}(f_i^0, f_j^0, f_{ij}^1) : i, j = 1 \ldots (n+m); i \neq j\} \\ 0 & \text{otherwise} \end{cases} \quad , l = 1 \ldots L \tag{1}$$

We assume that the resolution of the features is sufficiently fine, that the hash function is sufficiently good, and that L is sufficient large such that s features in an image would give rise to approximately distinct $s(s-1)$ combined features.

With these assumptions, we can make the following statements:

- each model feature vector contains $r(r-1)$ bits
- the image feature vector contains $n(n-1)$ bits that correspond to bits in the model feature vectors
- the image feature vector contains $m(n+m-1)$ spurious bits

[1] We consider combinations of two features only in this analysis; an analogous analysis goes through for combinations of three or more features

Our matching procedure consists of counting, for each model feature vector in the data base, the number of bits in the model feature vector that are present in the image. The model feature vector that has the largest number of correspondences wins.

It is straightforward to calculate the expected number of matches for each case:

- the correct model feature vector will have $n(n-1)$ certain matches with the image, plus $n(m+n-1)$ random matches against the remaining $r(r-1)-n(n-1)$ bits in the model feature vector.
- because of the assumed independence of model feature vectors, all other feature vectors will have $(n+m)(n+m-1)$ random matches against the $r(r-1)$ bits in each model feature vector[2].

In order to make a correct decision, we require that none of the incorrect model feature vectors have more matches against the image than the correct model feature vector. This is certainly satisfied if we require that all of the $N-1$ incorrect model feature vectors have less than $n(n-1)$ matches against the image feature vector with probability, say, $1-\epsilon$.

In essence, we are carrying out $N-1$ independent Bernoulli trials. Using the Poisson approximation to the binomial distribution, we require that in order to keep the likelihood of any success in the $N-1$ trials smaller than $1-\epsilon$:

$$\begin{aligned} 1-\epsilon &\geq e^{-\lambda} && (2)\\ \Leftrightarrow\ 1-\epsilon &\geq e^{-(N-1)p_{\text{trial}}} && (3)\\ \Leftrightarrow\ p_{\text{trial}} &\leq \frac{-\ln(1-\epsilon)}{N-1} && (4) \end{aligned}$$

For small ϵ, this is approximately:

$$p_{\text{trial}} \leq \frac{\epsilon}{N-1} \tag{5}$$

The probability p_{trial} is the probability that in any trial, more than $n(n-1)$ spurious features match the $r(r-1)$ features of a model. The probability that exactly k spurious matches occur is again given by a binomial distribution with $r(r-1)$ trials and probability of success of $\frac{(n+m)(n+m-1)}{L}$. Using Chebyshev's inequality, we obtain as a bound on the probability of obtaining more than $n(n-1)$ matches:

$$p_{\text{trial}} \leq \frac{r(r-1)(n+m)(n+m-1)}{Ln(n-1)} \leq \frac{r^2(n+m)^2}{L(n-1)^2} \tag{6}$$

If we use this result together with Equation 5, we obtain for the size of L:

$$L = \mathcal{O}(\frac{(N-1)r^2(n+m)^2}{\epsilon(n-1)^2}) \tag{7}$$

Therefore, the required size of the feature vector is at most linear in the size of the data base and the allowable error, and roughly quadratic in the number of features we expect in images or models.

The above has been a very simple statistical argument. We expect that the combined feature vector can be significantly smaller in practice, because our bounds have been loose, because essentially we have made a

[2] Actually, if the assumption of independence is not fulfilled, this fact will work for us rather than against us, since it is likely that mostly model feature vectors corresponding to the same object fail to be independent.

worst-case analysis, because we have used a very simplistic matching procedure, and because similarities among feature vectors belong to the same object model will work in our favor.

If we allow our system to make some number of mistakes b, the size of L decreases exponentially in b, i.e.:

$$L = \mathcal{O}(e^{-b}\frac{(N-1)r^2(n+m)^2}{\epsilon(n-1)^2}) \qquad (8)$$

This is of interest if we have an independent means of verifying matches and can tolerate a small number of potential false matches.

4 Conclusions and Discussion

The approach to visual model base indexing presented in this paper differs from other approaches in several respects. Probably the greatest point of departure from current approaches is that it does not make any explicit use of geometric facts in order to relate the locations of features in the image to the locations of features on the object. Instead, it makes enough of the structure of the problem explicit and accessible in the form of the combined feature vector that rather simple learning algorithms operating on the combined feature vector can solve the classification task well.

Our approach was motivated by the same information processing constraints that motivate other feature based object recognition algorithms–the viewing equation. However, as opposed to other approaches, it is robust with respect to deviations from this model, because it only takes advantage of the dimensionality and (piecewise) smoothness of the viewing transformation.

We believe that these key themes, using constraints that are robust with respect to model errors, using representations that make the structure of a problem easily accessible, and using simple learning algorithms, will be very important in many areas of machine perception. In fact, the current interest in "artificial neural network" algorithms is motivated by the observation that many algorithms in artificial intelligence and robotics make too strong assumptions about the structure of problems and as a consequence work only in well-delineated toy worlds (unfortunately, "artificial neural network" algorithms often err too far in the opposite direction and take advantage of no problem intrinsic constraints, resulting in impractical sample- and computational complexity).

It is, of course, still an open question whether *this* approach will generalize to realistic images. We have implemented a working system and applied it to the domain of wire frame objects; we have chosen this restricted domain because images are easy to generate and features are easy to extract from line drawings of wire frame objects. We are currently extending this work to arbitrary grey level images.

We feel that the empirical and theoretical results we have obtained so far look promising. In addition, we believe that our approach has relevance to psychological, psychophysical and neurobiological theories of object recognition.

Bibliography

Baird H. S., 1985, *Model-Based Image Matching Using Location*, MIT Press, Cambridge, MA.

Cass T., 1988, A Robust Implementation of 2D Model-Based Recognition, In *Proceedings IEEE Conf. on Computer Vision and Pattern Recognition*, Ann Arbor, Michigan.

Goldberg R., Lowe D., 1987, Verification of 3D parametric models in 2D image data, In *Proceedings of the IEEE Computer Society Workshop on Computer Vision (Cat. No.87TH0210-5).*

Grimson W. E. L., Lozano-Perez T., 1985, Recognition and Localization of Overlapping Parts from Sparse Data, Technical Report A.I. Memo 841, MIT.

Grimson W. E. L., 1984, The Combinatorics of Local Constraints in Model-Based Recognition and Localization from Sparse Data, Technical Report A.I. Memo 763, Artificial Intelligence Laboratory, Massachusetts Institute of Technology.

Grimson W. E. L., 1988, The Combinatorics of Object Recognition in Cluttered Environments using Constrained Search, Technical Report A.I. Memo 1019, Artificial Intelligence Laboratory, Massachusetts Institute of Technology.

Hoffman D. D., Richards W. A., 1983, Parts of Recognition, Technical Report AIM-732, Artificial Intelligence Laboratory, Massachusetts Institute of Technology, Cambridge, Ma.

Huttenlocher D., Ullman S., 1987, Recognizing Rigid Objects by Aligning Them with an Image, Technical Report AI-Memo 937, Artificial Intelligence Laboratory, Massachusetts Institute of Technology.

Lamdan Y., Wolfson H. J., 1988, Geometric Hashing: A General and Efficient Model-Based Recognition Scheme, Technical Report Technical Report No. 368, Robotics Report No. 152, New York University, Robotics Research Laboratory, Department of Computer Science, New York, NY.

Lowe D. G., 1986, *Perceptual Organization and Visual Recognition*, Kluwer Academic Publishers, Boston.

Lowe D., 1987, The viewpoint consistency constraint, *Int. J. Comput. Vision (Netherlands)*, 1(1).

Marr D., 1982, *Vision: A Computational Investigation into the Human Representation and Processing of Visual Information*, W.H. Freeman and Company, San Francisco.

Shirai Y., 1981, Use of models in three dimensional object recognition, In Sata T., Warman e., ed.s, *Man-machine communication in CAD/CAM. Proceedings of the IFIP wg5.2-5.3 working conference.*

Optimization of 2-Dimensional Model Matching [1]

J. Ross Beveridge R. Weiss E. M. Riseman

March 30, 1989

Abstract

A methodology for identifying known 2D models in broken and skewed line data is presented. Such data requires matching techniques that involve one-to-many mappings between model and data line segments. The relevance of this problem to 3D vision, in particular robot navigation, is demonstrated. At the heart of the matching process is an effective measure of spatial fit for which a closed form solution is given. The problem of establishing the correct correspondence between model and data lines is cast in terms of combinatorial optimization, and local search is shown to be effective. A generalized Hough transform for determining model translation and rotation parameters is used to determine promising initial matches for the local search process. Experimental results on complex outdoor scenes are included.

1. Introduction

A central problem in computer vision is the identification of objects in the world via features derived from digital images. This often involves matching an object model to data extracted from an image. For rigid objects the model might be a 3D construct, for example a wire frame. The identification task has two parts: a) determining the correct correspondence between object features and image features and, b) determining the position of the object with respect to the camera. These sub-tasks are interdependent, since an object's position cannot be determined without assuming a correspondence to image features, while the correct correspondence depends on the object's 2D appearance and hence its relative position and orientation in space.

1.1 Model Matching and Optimization in Two vs. Three Dimensions

We believe that there are strong incentives to solve as much of the identification problem as possible via processing in the 2D image space. The combinatorics of establishing correspondences between object and image features dominates the identification problem, and geometric computations integral to this process are simpler in 2D than in 3D. In particular, this paper shows that the determination of the optimal position of an object's 2D projection with respect to corresponding image features has an analytic solution in the two dimensions of image space. This closed form solution is a new result and we believe it to be a significant contribution. It is highly doubtful that the related 3D problem has an analytic solution for determining model positions that minimize point-to-line and point-to-plane distances.

The reader should note that if point-to-point correspondences between model and image exist, there are analytic solutions for optimization in both the 2D and 3D domains [Hor87]. However, point-wise correspondences are generally very difficult to obtain in a reliable manner. When line features are extracted, the location of their endpoints is quite prone to error. A typical technique, in fact, is to use vertices of line pairs to more accurately determine point locations. However, such a strategy does not always work, and one can see examples in our experiments later in this paper where line correspondences are possible but point correspondences are not feasible. In this paper we assume that line tokens can be reasonably extracted from an image (with some errors of course), but that 2D points often cannot be.

The combinatorial complexity of establishing a correspondence between object features and image features would be high even if it could be assumed that for each and every object feature there existed a single corresponding image feature. However, when the best straight line algorithms are applied to natural complex scenes, the output almost always involves fragmented, overextended, and missing lines. Hence, the correct correspondence often involves complex processing, for example associating several fragmented image line segments with a single model line. This amounts to integrating the grouping of line fragments into the model matching task, thereby further adding to the combinatorial complexity of the task.

Determining the 3D position of 3D models can be decomposed into a 2D matching problem followed by 3D pose refinement. This is made easier when a rough estimate of the object position is available. Autonomous navigation

[1] This work was supported by the Defense Advanced Research Projects Agency under grant F30602-87-C-0140 and by the National Science Foundation under grant DCR-8500332.

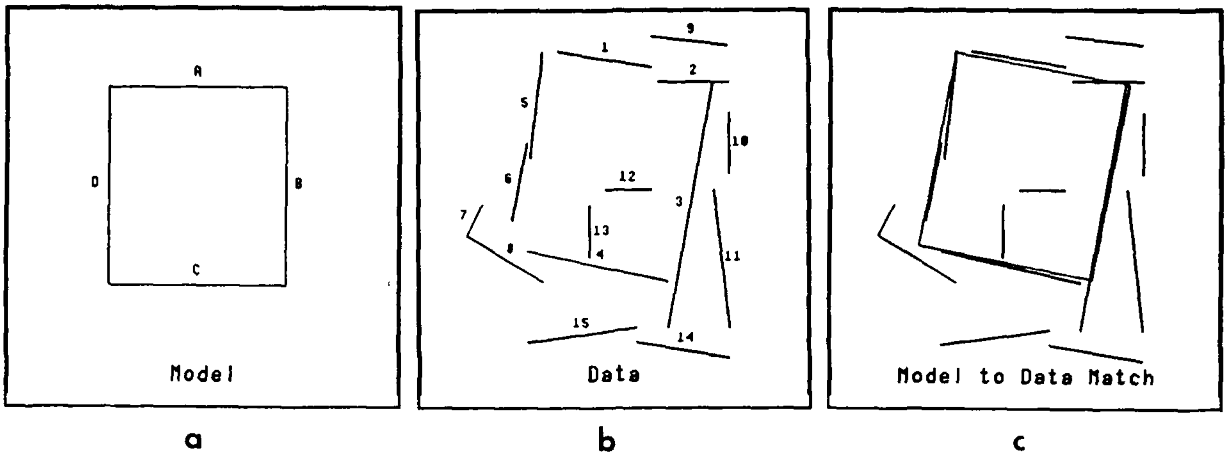


Figure 1: An example of the type of problem we wish to solve: a) a model of a rectangle, b) broken and skewed data of the kind produced by bottom-up algorithms on complex imagery. c) the desired match between model and data.

of a robot via landmark recognition is just such a problem and represents an important application of the methods developed in this paper. The robot will update its estimated position by determining its relative position with respect to the visible landmarks. The best way of estimating the 3D position of the camera (and hence robot position) given the image lines corresponding to 3D landmarks is itself a research question. A robust method for performing this task is presented in [Kum89]. In this paper identifying the landmarks involves matching data line segments from the image to 2D projections of these landmarks. Unless errors in the expected 3D position of the robot introduce substantial 2D distortions in the landmark projection, this approach is reasonable. Presuming that a robot planner is selecting which landmarks to observe it can generally select landmarks for which distortion will be minimal. For more details on the integration of 2D landmark recognition and 3D navigation see [FHR89].

It should not be forgotten that some visual tasks are inherently or largely two-dimensional in nature. For such tasks the methods presented here are directly appropriate. For example, distant aerial interpretation, although not strictly a 2D problem domain, can often be treated as such; the viewing angle is constrained and objects are always far from the camera. The matching techniques presented here may also be useful for certain industrial automatation tasks.

1.2 Overview of the Correspondence and Spatial Fitting Problems

The type of 2D matching problem to be solved is illustrated in Figure 1. In the remainder of this paper the term "model" will typically refer to a "2D line model" such as the rectangle in the Figure 1a, although in general the model will be the projection of a 3D object as we have just described. Models will be assumed to be rigid, although there is no assumption that model line segments must form a closed polygon. The term "data" will refer to "data line segments" extracted from a digitized image by some low-level algorithm as illustrated in Figure 1b. The problem, in short, is to match the model to the data as illustrated in Figure 1c. The current example has been hand drawn for the sake of illustration. Data derived from images will be presented along with results at the end of the paper.

The types of 'errors' in the line data shown in Figure 1 are illustrative of what can be expected from bottom-up algorithms such as [BHR86]. Data lines are often fragmented; they may extend beyond or fall short of the point predicted by the model, and often they are skewed. These discrepancies between model and data are a direct consequence of several factors. In cluttered, unconstrained domains such as outdoor scenes, ambiguity in the data is inevitable and hence model and data will rarely match perfectly. Sources of ambiguity include but are not limited to variations in lighting, occlusion, and coincidental structure such as alignment between distinct objects. Moreover, by necessity object models are simplifications of reality, hence unmodeled structure will always be another source of discrepancy. Probably the most significant factor is the limitations and idiosyncracies of line extraction algorithms.

Given that matches will seldom if ever be perfect, the emphasis must be on determining the 'best' of the imperfect matches. Hence matching is naturally posed in terms of optimization over the possible matches. By establishing an objective measure of match quality, the problem becomes one of determining the correspondence between model elements and data line segments for which the measure is optimal. The correspondence problem is combinatorial, and generally involves mapping one model line to many data lines. A second optimization problem is *implicit* in the correspondence problem. In order to measure the quality of a given set of model-data line correspondences, the best 2D position of the model with respect to the data must be determined, and the extent to which they do not spatially coincide must be measured. This we will call the *fitting* problem. Thus, a *match* involves both model-data correspondence and an associated best-fit position. A change to the correspondence will, as a rule, change the optimal fit between model and data, and therefore the model's position. Thus, for every possible correspondence between model and data, there is an associated best fit postion. There is also a strong connection in the reverse direction. The proper position of a model might be roughly constrained via knowledge (e.g. the location of the robot position and orientation), or via some intermediate processing technique as we shall discuss shortly. If the location of the model is roughly known, there are a limited number of reasonable correspondences; i.e. only those data lines that are "near" the expected position of each model line would be considered as possible correspondences.

The following is a sketch of the basic steps used to obtain a good model match.

- Determine the search space of correspondences. Lacking constraints on model position, all data lines segments possibly correspond to every model line segment. If constraints are available, only associations of model and data lines satisfying these constraints need be considered.
- Determine promising model positions if the search space is large. Use these positions to determine constrained search subspaces made up of only correspondences consistent with the estimated position. A promising model position may be found either through a generalized Hough transform or by identifying prominent features. The generalized Hough technique involves an analysis of the space of possible two-dimensional spatial transforms to bring the model and data into alignment. Identifying a prominent feature may involve finding a distinctive part of a model such as a corner, and then using that to position the model as a whole.
- For each of the constrained search spaces (sets of possible model-data correspondences) obtained above, use iterative refinement to determine a best match. Upon each iteration perturb the correspondence, adding or deleting one or several data lines, and then determine the new best-fit model position and related match error. If the match error is reduced adopt the improved match; stop when the match can no longer be improved. The best of the resulting matches is taken as the final match.

To review our use of terms: a "match" is a correspondence whose spatial fit has been optimized; a "globally optimal match" is the correspondence in the search space which has the lowest match error; and a "locally optimal match" is a correspondence which is optimal in a localized neighborhood of the search space of correspondences. Generally "best match" will refer to a locally optimal match that is likely, but not guaranteed, to be globally optimal.

After a brief survey of related work on line-based model matching, we will turn to the problem of finding the best match. This will involve a standard technique from combinatorial optimization called *local search*. Next the objective function used in 2D matching is presented. Special attention will be drawn to the problem of obtaining the appropriate measure of spatial fit between model and data. We show that our choice of fit measure is superior to the reasonable alternatives, and we present for the first time an analytic solution for optimizing model placement using this measure. This analytic solution is particularly valuable since searching for the best match involves fitting the model to the data for a number of possible correspondences. Since finding the best match is expedited by a good initial estimate of model position, we will discuss briefly how to obtain such estimates. Finally, results will be presented for robot landmark identification. Matches for this domain have been used to reliably determine robot position over a sequence of six images. These results are presented in these proceedings [Kum89]. One additional result illustrates the strength of our technique at overcoming even dramatic weaknesses in the bottom-up line data.

2. Work Related to Line Based Model Matching

Object recognition can be thought of in two complementary ways. In one way, recognition means finding a correspondence between related parts of an object in the model base and related features in an image description, a correspondence that satisfies the constraints entailed by the structure of the object and the geometry of image formation. Methods based on this point of view rely on distinctive features that characterize each object, as exemplified by the local-feature-focus method of Bolles and Cain [BC82]. The other approach is to find the parameters of the transformation that maps an object into its appearance in an image. Methods based on this point of view combine weak evidence from many features, as exemplified by the generalized Hough transform (such as Ballard and Sabbah [BS83], Silberberg *et al.*[SHD84]).

The feature focus method is based on matching a set of features starting with one that is distinctive. A match of the initial feature focusses the search to matching nearby features. The largest maximal clique of correspondences between model and image features is used to generate a hypothesis for the transformation. The distinctiveness of the

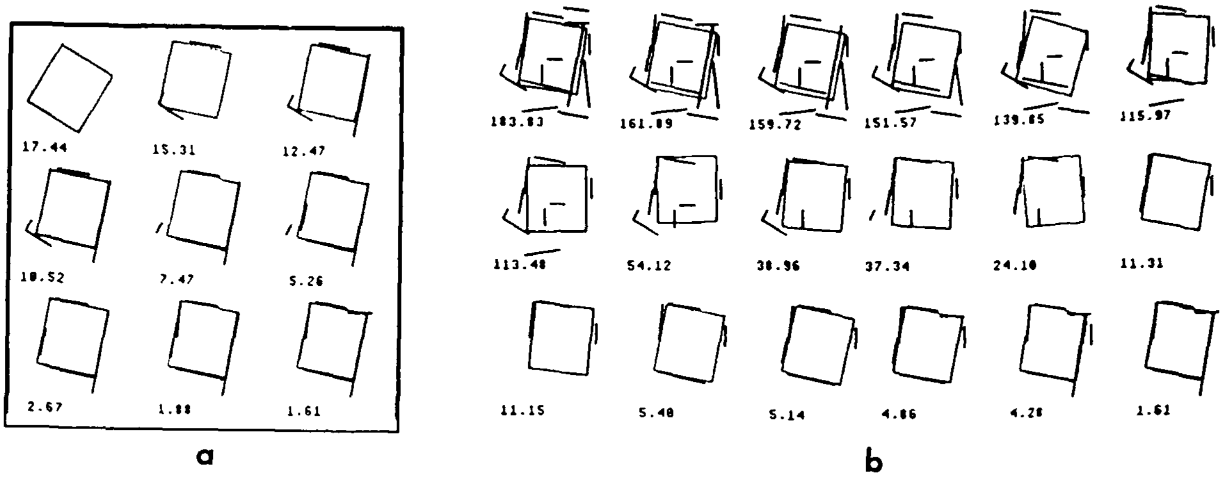


Figure 2: Two distinct sequences leading to the sam.e optimal match: a) starting from a corner and, b) starting from a 'full' match, all lines found near the model in its estimated (in this case correct) position. The sequences should be read left to right, top to bottom.

features is important for limiting the size of the graph when searching for cliques. Aside from prediction from model projection, another method for extending a match is perceptual organization. This makes it possible to form complex features that are more distinctive for each object. The SCERPO system by Lowe [Low85] follows this approach. In addition, as in ACRONYM [Bro81,Bro82], this algorithm alternates between finding more correspondences and updating the transformation. An issue that needs to be addressed is whether the reduction in the search space by using distinctive features offsets the computational complexity of forming them.

A major problem with these methods is the exponential complexity of the search over all possible correspondences. Goad [Goa83] does an analysis that permits pruning of the search tree for recognizing an object, based only on the geometry of the object itself and the general constraints of the imaging geometry. This analysis can be compiled ahead of time, before any images are interpreted. The work of Grimson and Lozano-Pérez [GL84], in the related area of object recognition by tactile sensing, is similar in intent.

The Hough transform is another method for eliminating an exhaustive search of all correspondences and has a generalization implemented by Tucker *et al.* [TFF88] on the Connection Machine. Each image-model feature pair votes for a model-to image tran' :ormation. This requires that the feature pair determine the transformation uniquely; this is easier for two dimensions than it is for three. Each transformation is a hypothesis, and using the entire model, the hypotheses with the most votes are verified first. Thompson and Mundy[TM87] developed a parallel algorithm for recognizing 3D objects using vertex pairs. This algorithm proved to be very efficient because it used only pairs of features to compute the 3D pose of an object. The clusters or peaks are found in a 2D subspace of the 6D parameter space. Another variation on the Hough transform is geometric hashing. This method was used by Schwartz and Sharir [SS85] and Kalvin [KSS86] for 2D matching and described more generally by Wolfson and Lamson [LW88].

3. Local Search as a Means of Finding Best Matches

Local search is a promising means of establishing the optimal correspondence between model and data. A formal introduction to local search may be found in [PS82]. Local search offers perhaps the most practical means of solving many NP-complete problems, such as the Traveling Saleman Problem [LK73]. The process of local search, put simply, is an iterative generate-and-test procedure which moves from an initial solution via transformations to one that is locally optimal. Note that the common notions of hill-climbing and iterative refinement are in essence local search. Formally characterizing our 2D matching problem as a combinatorial optimization problem allows us to utilize this general methodology. To do this we will define the objective function, the search space, the neighborhood, and the search strategy.

Before moving to general definitions however, let us illustrate with an examnle how local search is applied to 2D matching. Referring back to F'gure 1 note that the model and data lines are ui quely identified by letters and numbers respectively. For the matcn shown in Figure 1c, the implied correspondence or mapping between model and data is: $A \leftrightarrow \{1\ 2\}$ $B \leftrightarrow \{3\}$ $C \leftrightarrow \{4\}$ $D \leftrightarrow \{5\ 6\}$. This is the intuitively obvious match, and as we are about to see, it is also the optimal match.

PAIRINGS, MODEL TO DATA LINE SEGMENTS															
	A			B			C					D			
STEPS	1	2	9	3	10	11	4	8	12	14	15	5	6	7	13	EVALUATION
Search Path from Partial Match																
Initial								√						√		17.44
1	√							√						√		15.31
2	√			√				√						√		12.47
3	√			√			√	√						√		10.52
4	√			√			√							√		7.47
5	√			√			√						√	√		5.26
6	√			√			√						√			2.67
7	√			√			√					√	√			1.88
final	√	√		√			√					√	√			1.61
Search Path from Full Match																
Initial	√	√	√	√	√	√	√	√	√	√	√	√	√	√	√	183.83
1	√	√		√	√	√	√	√	√	√	√	√	√	√	√	161.89
2	√			√	√	√	√	√	√	√	√	√	√	√	√	159.72
3	√				√	√	√	√	√	√	√	√	√	√	√	151.57
4	√					√	√	√	√	√	√	√	√	√	√	139.85
5	√					√	√	√	√		√	√	√	√	√	115.97
6	√					√		√	√		√	√	√	√	√	113.48
7	√					√		√	√			√	√	√	√	54.12
8	√					√		√				√	√	√	√	38.96
9	√					√						√	√	√	√	37.34
10	√					√						√	√		√	24.10
11	√					√						√	√			11.31
12	√					√						√				11.15
13	√					√	√					√				5.40
14	√					√	√					√	√			5.14
15	√	√				√	√					√	√			4.86
16	√	√		√		√	√					√	√			4.28
final	√	√		√			√					√	√			1.61

Table 1: A trace of the local search process for the two parts of Figure 2. A √ designates a correspondence between a model line and a data line.

Figure 2 shows two distinct match sequences, each terminating with the optimal match. The sequence in Figure 2a starts with the model matched to two data line segments which form a corner, as shown in the upper left hand portion of the figure. Here the corner may be thought of as a key feature used to establish the initial position of the model. However, note this particular corner is not part of the best match. The starting match in Figure 2b might be the result of estimating model position, perhaps with a generalized hough transform, and then establishing a correspondence between model and data line segments which are near each other and properly aligned.

The model, drawn in grey, is always shown in its optimal position with respect to the data. The result of applying the objective function is written below the match. Details about the choice of model position and objective function will be described in following sections; here our focus is on the local search process. Each successive match in these sequences is an improvement over the previous match. Each sequence terminated when no additional improvement was possible, and in each case with the same optimal match.

In both sequences successive matches differ by only one model-data line correspondence. In these illustrations the exact choice of data line to add or delete in the correspondence set was a function of the order in which they were tested, since the first change for which there was an improvement in the objective function was adopted. Alternatively, the line for which there was the greatest improvement could have been selected. This distinction is reflected by two of the most common strategies for local search, first improvement and steepest descent. Had steepest descent been used (i.e. choose the change in model-line correspondence for which the improvement is greatest), the succession of improved matches might have led more directly to the best match. However, when there are many possible changes to the correspondence set to be examined, steepest descent can turn out to require significantly more tests than first improvement. For this reason we favor first improvement. In the future we plan to conduct a thorough comparison of these two strategies.

Table 1 traces the correspondence between model and data for matches in Figure 2, and provides insight into our search space. In general, if there are M model line segments and L data line segments, then there are $N = L \cdot M$ possible pairings between model and data line segments. This representation permits a many-to-many mapping,

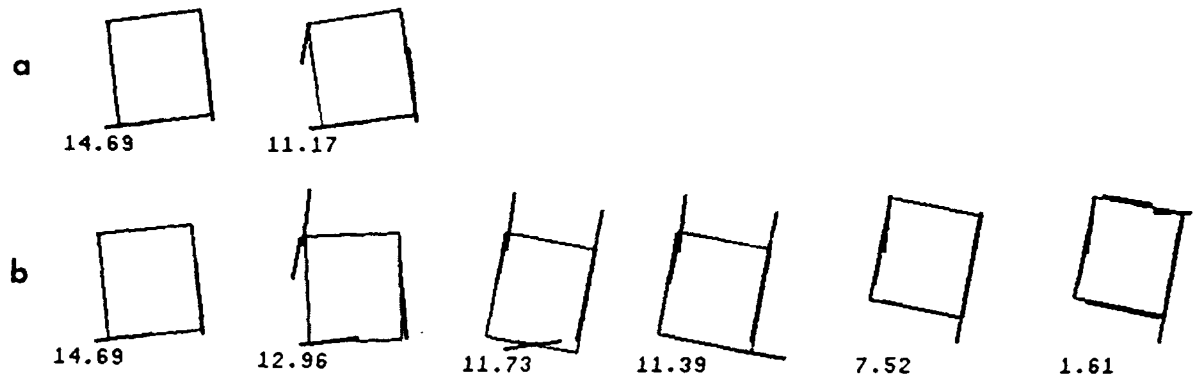


Figure 3: A local minimum for a $k = 1$ add/delete neighborhood disappears for $k = 2$. a) A first improvement search for $k = 1$ finds a local minimum after one tranformation. b) A firs: improvement search for $k = 2$ from the same initial match finds the optimal match.

and the typical case is one model line mapping to two or more data lines. Mapping one data line to more than one model line is penalized by the objective function, and could be precluded from the search space altogether. When the model position is roughly known, then only data lines in correspondingly appropriate positions need to be considered. Hence, as illustrated in Table 1 , N may be considerably smaller than $L \cdot M$. For example, $A \leftrightarrow \{6\}$ is not a possible correspondence in Table 1 .

In local search the neighborhood defines those matches which are related. Here, conceptualizing the search space as a graph is helpful. Each correspondence in the search space is a node. A link is placed between nodes which differ by a single *transformation*. Nodes with a link to a given node form its neighborhood. For example, consider a transformation operator which adds or deletes k data lines from the model-data line correspondences. The neighborhood of the correspondence of a given match consists of all matches which differ by k data lines.

Observe that a locally optimal match using one neighborhood definition may not be locally optimal if the neighborhood is expanded. The neighborhood definition used in the above illustration corresponds to a k add/delete transformation with $k = 1$. In common practice the $k = 1$ neighborhood has undesirable local minima. Such a local minimum is shown in Figure 3a, in which B is initially matched with 11 and C is initially matched with 15 (using labels from Figure 1). Using the $k = 1$ add/delete transformation leads to a local minimum after adding $D \leftrightarrow \{5\}$ to the match correspondence set. In Figure 3b, using $k = 2$ for the same initial match and search strategy leads to the optimal match. Thus, the local optimum with $k = 1$ disappears for $k = 2$.

Obviously larger k values are better; however, the neighborhood size using the k add/delete transformation is $O(N^k)$; the increase in computational cost associated with increasing k is dramatic. Our experience suggests that a first improvement strategy on a $k = 2$ neighborhood usually leads to a globally optimal match when used in conjunction with intelligently selected initial matches. More complete statistical studies will be reported in future presentations.

The 2D matching problem differs from problems like the the traveling salesman problem in one significant respect. When two cities swap position in a tour of the traveling salesman problem, the change in cost can be computed from just those links associated directly with the change. There is no need to evaluate anew the entire tour. However, any change to the correspondence in our 2D matching problem requires a spatial repositioning of the model with respect to the data. Consequently, the contribution from every pair of model and data lines to the overall match quality in terms of fit error and omission (see Section 4.) must be recomputed based on this new position.

Heuristic tests may circumvent the need to recompute the global evaluation under certain conditions, and these may represent valuable and pratical shortcuts, but the matching problem is by definition inherently global if model-data line correspondences are formally included in the definition of the optimization problem. On the positive side, repositioning of the 2D model with respect to the data requires only rotation and translation in the image plane. One of the contributions of this paper is an analytic solution to this optimization problem. With our current implementation it is practical to test hundreds or thousands of possible matches, but not more. Hence, there is still strong incentive to limit the neighborhood size and to start with relatively good initial matches, a topic that we will return to in Section 5. At this point note that the global character of matching both encourages careful choice of

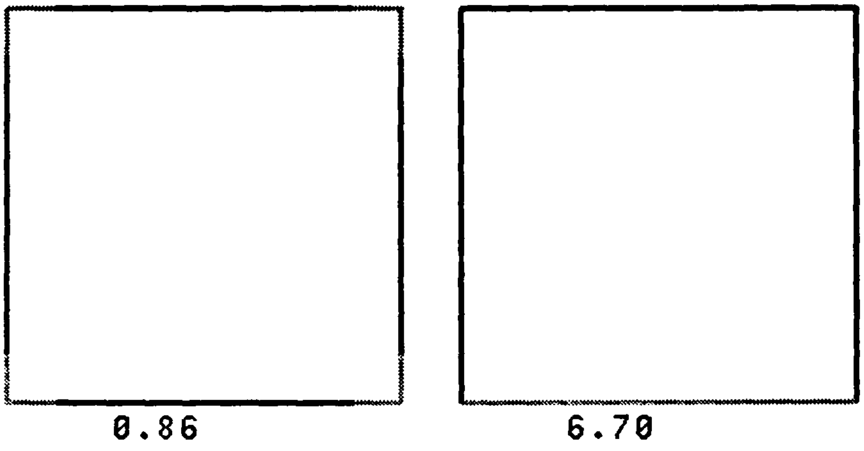


Figure 4: The total percent omission taken over all four model lines is the same for each match. However, since we consider the lefthand match to be better, the value of the omission error is considerably lower than for the righthand match.

initial matches and provides a general means for obtaining them.

4. The Objective Function – Match Error

The objective function is the formal evaluation measure defining the quality of a match. No single, universal objective function exists for all matching problems. The objective function might have to be varied or replaced entirely to satisfy the dictates of a specific application. However, our objective function is general, practical, and we believe appropriate for most 2D matching problems. It is based on the observation that a good match is one where every portion of every line in the model is accounted for by data. Figuratively speaking, the model should be able to be placed spatially over the data, and everywhere there is a model element there should be corresponding data. A match can fail to meet this objective in two distinct ways: data lines may be displaced and skewed, or portions of the model may be missing. For each line, the objective function or *match error* is defined as:

$$E_{\text{match}}(l) = E_{\text{fit}}(l) + \alpha E_{\text{omission}}(l)$$

where E_{fit} measures the spatial fit between the model and corresponding data, and E_{omission} measures how much of the model is missing from the data. A perfect match would yield an E_{match} of zero.

E_{match} for the model M is a sum of weighted errors, each associated with an individual model line segment l:

$$E_{\text{match}} = \sum_{l \in M} w_l E_{\text{match}}(l)$$

This decomposition facilitates alternative choices of weights w_l; two obvious alternatives are equal weight for all model segments or weight proportional to length. To fully justify the choice of one over the other requires recourse to the goals of the specific matching problem and the characteristics of the task domain. In recognizing navigational landmarks, small line segments may be just as important as large ones; e.g. the short top crossbar of a telephone pole is critical to distinguish among the vertical lines that are common to telephone poles, lamp posts, and some tree trunks. Hence, we choose equal weighting for the task domain described in this paper.

The approach of attributing error by individual model segment is appropriate for dealing with omission of data in evaluating matches. For each model line segment E_{omission} will be defined to be a non-linear function of the percent of the line that is unaccounted for by data. The measure is constructed in this manner because even under the best of circumstances a small amount of omission is to be expected (e.g. the ends of lines are often difficult to extract). However, if large portions of a model segment are missing from the data, the estimated quality of the match should be substantially reduced. Consider the two alternative matches illustrated in Figure 4. Due to the nature of bottom-up line extraction, the four sided match is preferable, even when the total percentage of model line length is somewhat less. Letting P be the percent of the model line l omitted by the data, $E_{\text{omission}}(l)$ is defined as:

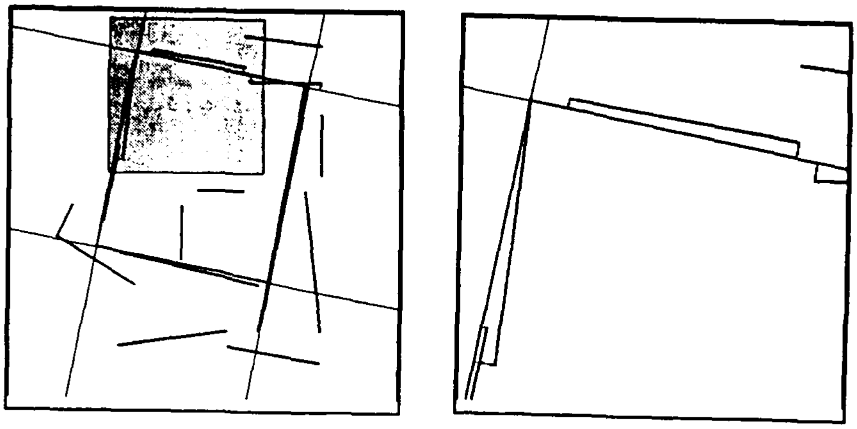


Figure 5: Lateral displacement in model projection is illustrated for the best match from the previous example. Note that model lines are infinitely extended, and the perpendicular distance from the end- points of data line segments to model lines is minimized. Part b) shows an enlargement of the shaded area.

$$E_{\text{omission}}(l) = e^{\beta P}$$

This formulation expresses the increasing importance of larger omission values. The change in omission is proportional to the omission, specifically:

$$\frac{dE_{\text{omission}}}{dP} = \beta E_{\text{omission}}$$

Proper choice of the proportionality constants of the objective function, α and β, is of course an issue. Empirically selected values for α and β along with the resulting best matches are presented below. For a given domain, such as landmark recognition in our outdoor campus, one set of constants is used for all models. If specific tuning were required for each model, the value of our approach would be undermined. However, there is every indication that one choice of constants performs effectively for many models in a given domain. Nevertheless, some domain dependency may be expected. For example, changes in the nature of the errors present in the line data may dictate ᵖ ·hange of α, which balances the tradeoff between fit and omission.

4.1 Optimization of the Spatial Fit of the Model

For a given set of model-data line correspondences, the error for spatial fit contributing to the objective function should be derived from the model placed in the best possible position. Consequently, the spatial fit must be optimized separately for every set of model-data line correspondences that is considered. Therefore, the computational form for determining the optimized fit is of great importance, and only quadratic fit measures have been considered.

Simple quadratic measures of point-to-point distances for this problem are inappropriate. As we discussed earlier, line segments produced by bottom-up algorithms will be fragmented and partially missing; therefore, a direct mapping of model end-points to data end-points is impossible. To further complicate matters the exact point where a line terminates is unreliable. Even if the two data line end-points most closely associated with a model line end-point were identified, it would still be inappropriate to take Euclidean point-to-point distance as the measure of fit. A far more effective approach is a point-to-line measure. The need for a point-to-line measure was identified clearly by Lowe [Low80,Low85].

The most reliable quadratic point-to-line measure is *model-projected lateral displacement*, which is a weighted sum-of-squared perpendicular distance from the end-points of the data lines to the corresponding model line. This measure is illustrated in Figure 5, where the best model match from the earlier example is shown with the extended model lines explicitly drawn, and the projections of the end-points of data line segment onto the extended model lines also explicitly drawn. In the enlargement the projection can be easily seen.

E_{fit} *is formally defined as:*

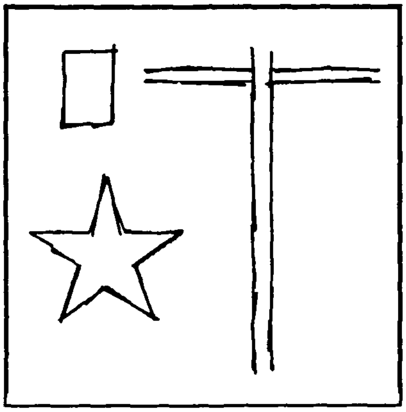


Figure 6: The three models in the study of alternative fitting methods. The data shown is the result of the stochastic breaking and skewing process.

$$E_{\text{fit}} = \sum_i \ell_i \left(n_i \cdot (R(p_i) - T) - \rho_i\right)^2 \tag{1}$$

Each term in the sum denotes the perpendicular distance from data line end-point p_i to the corresponding model line l_i. Note that l_i has unit normal n_i, and its distance from the origin in the direction n_i is ρ_i. The contribution from data line end-points is weighted by the length ℓ_i of the data line segment.

The problem of determining the 2D rotation and translation parameters which minimize E_{fit} has an analytic solution. The translation which minimizes E_{fit} for a given angle of rotation ϕ may be expressed as a linear system of equations (B and V are both constant):

$$T = B\hat{R} - V \qquad \hat{R} = \begin{vmatrix} \cos\phi \\ \sin\phi \end{vmatrix} \tag{2}$$

We substitute the optimal T back into equation 1 and differentiate with repect ϕ, obtaining a fourth order polynomial in $\cos\phi$: One of the roots of this quartic equation corresponds to the rotation which minimizes E_{fit}. The roots of the quartic can be solved for directly. We believe this closed form solution represents a significant contribution to symbolic model matching, and its complete derivation is included in Appendix A. It is worth noting that others in the literature found iterative methods necessary for solving the related 3D problem [Low85,LHF88].

A stochastic model of the breaking and skewing process has been used to compare model-projected lateral displacement to two alternative fit measures. The two alternative measures were selected because they represent previously published methods. The first alternative is data-projected lateral displacement and has been used by Lowe [Low85]. As in our model-projected lateral displacement, Lowe's measure uses a point-to-line distance. However, it reverses the roles of model and data. Hence, the sum-of-squared perpendicular distances from end-points of model line segments to data lines is minimized. It can be shown that this lateral displacement is less reliable than our model-projected measure. Consider a model line segment matched to a data segment that is only half its length and is skewed slightly. Extending the data line and measuring lateral displacement from the ends of the model line segment to the extended data line amplifies the error introduced by skewing. In contrast, measuring distance from the end-points of the data line segment to the extended model line will not suffer from this problem. Since skewed data lines are a fact of life, this turns out to be a rather important difference.

The second alternative measure considered here minimizes the sum-of- squared point-to-point distances between corresponding model and data vertices. A model vertex is the point at which two adjacent model lines intersect. This type of vertex approach is essentially what is used by [TFF88] and [HU87]. For testing this measure on our data vertices were specified by hand. Even with the properly selected vertices, this method is less reliable in our experimental tests than model-projected lateral displacement. Of course these results are a function of the fragmentation and skewing that takes place in any particular straight line extraction algorithm, and to better understand them more must be said about the experiment.

A statistical study of the alternative fit measures was performed using a stochastic model of the breaking and skewing process. Three distinctive geometric shapes were selected: a rectangle, a star, and a telephone pole. The

	RECTANGLE, 1000 trials		STAR, 100 trials		POLE, 100 trials	
Method	Mean	Standard error	Mean	Standard error	Mean	Standard error
1	0.2148	0.0036	0.1659	0.0063	0.1088	0.0067
2	0.3397	0.0057	0.1869	0.0074	0.7600	0.0403
3	0.2362	0.0034	0.2898	0.0182	1.3781	0.0967

Table 2: Comparison of three different fitting measures on three models shown in Figure 6. For each measure, the optimum position of the model with respect to noisy data is found. The distance between this estimated position and the correct one is computed. The mean and standard error is shown for a large number of trials, and the mean for measure 1 is significantly less than for the others.

perfect model segments were randomly broken and skewed. An illustration of the models and corresponding data is shown in Figure 6. The three distinct fitting methods were then used to position the model with respect to the corrupted data and the deviation from the true position of the model was measured using the distances between corresponding vertices. Note that an ideal measure of fit would be insensitive to this type of data corruption, and hence would not change the model position. We found that our model-projected lateral displacement came closest tc this ideal. The -esults over a large number of random trials are summarized in Table 2. It is worth noting that the vertex method gave comparable results for the rectangle, but degraded seriously for the star and pole.

5. Generating Initial Matches

There are two basic approaches to the problem of finding good initial matches: a) Finding characteristic substructure, i.e. key features, which indicate the presence of the model ([BC82,Low85]); b) Use of a generalized hough transform which relate model and data ([Bal81]).

Effective heuristics for identifying key features depend on characteristics of the models and the data. If a model contains a prominent corner, then searching for corners in the data makes sense. The common theme is that a portion of the model that is sufficient to uniquely determine position is identified in the data. The model position is determined from this partial match, and this in turn restricts the additional data that can participate in the match. The approach is effective when key features are readily and reliably identified, but when they are not, substantial amounts of computation can be wasted.

Generalized hough tranform techniques involve voting in a transformation space. Correspondences of model and data primitives, for instance pairs of model and data lines, independently indicate support for a particular transformation or family of transformations. The generalized hough is a global histogram that represents the accumulation of the local information, and peaks represent globally interesting transformations. The transformation spaces and primitives vary across applications. For the matching of 2D rigid models, the natural transformation space has three dimensions. The axes are: u - the translation in the x direction, v - the translation in the y direction, and ϕ - the rotation in the plane.

The choice of the appropriate primitive is less obvious. It has been common practice to select primitives for voting that uniquely determine a point in the transformation space. For 2D matching a correspondence between two points or a point and a line is necessary to uniquely determine rotation and translation between model and data. For example, the system presented in [TFF88] used pairs of lines forming a vertex as the primitive. Such primitives, like key features, must possess sufficent structure to uniquely determine model position. However, as our implementation illustrates, less constrained primitives may be used in a generalized hough transform.

Our primitive is a pair, one model line and one data line. The family of transformations associated with such a pair is quite intuitive. To the extent that a given transformation places the model line segment over the data component, that transformation is favored. The underlying idea, in keeping with our matching objective function, is that the best match position should have the model "on top of" as much of the data as possible. This approach has proven quite effective. Note that our choice of a primitive facilitates matching even when extreme fragmentation and skewing cause significant difficulty for any sort of vertex-based approach. An illustration of matching to extremely fragmented data will be presented in the next section. For more details on our use of the generalized hough transform see [BWR89].

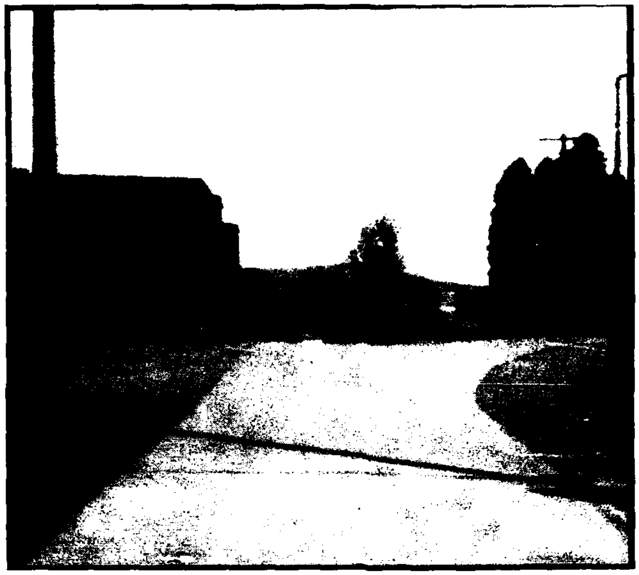


Figure 7: A 512x512 image from the robot domain

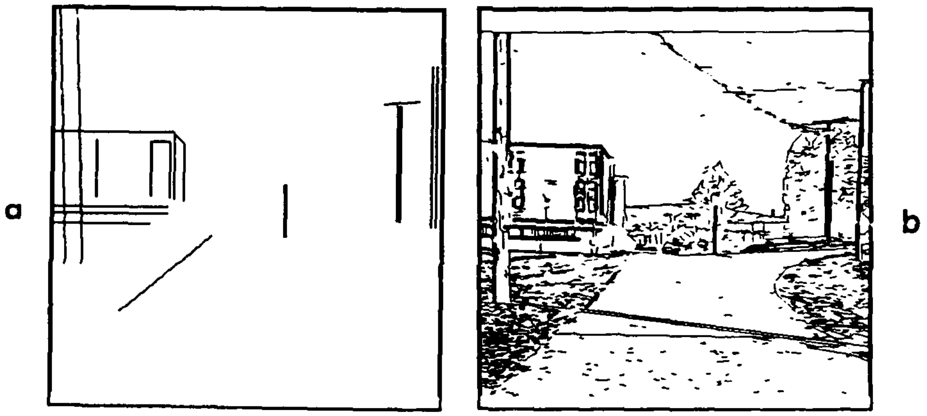


Figure 8: a) The six navigational landmarks projected onto the image plane. b) Data line segments matching the landmark lines.

6. Model Matching Results

A system implementing the matching methods described above has been built on a T.I. Explorer II Lisp Machine. 'The previous illustrations were all generated by this system. The bottom-up data line segments for this system are generated by the Burns line algorithm [BHR86] and are then filtered by length and contrast.

For the robot's outdoor test area we have successfully demonstrated the updating of 3D robot position via landmark identification followed by 3D pose refinement. A sequence of six 512x512 images were acquired. The first of these images is shown in Figure 7. Our experiment involved projecting 3D landmarks onto the image plane using an estimate of the robot's position. The landmark projections were then matched to bottom-up data. Using the resulting matches the robot's position was recovered to within about a foot for each of the six images. The specific results for the pose refinement portion of this experiment are presented in [Kum89].

The landmark projections for the first image are shown in Figure 8a. The six landmarks are: two telephone poles, a lamp post, a street lamp, the side of the walkway and the building. These projections were assumed to be correct to within 30 pixels and 0.3 radians. All correspondences of single model line to single data line that satisfy these constraints voted into the transformation space, and the largest peak in the space was used to initially position the model. From this position, a restricted space of correspondences was formed; the same constraint was used again so that only pairs of model and data line segments within 3 pixels and 0.3 radians of each other were

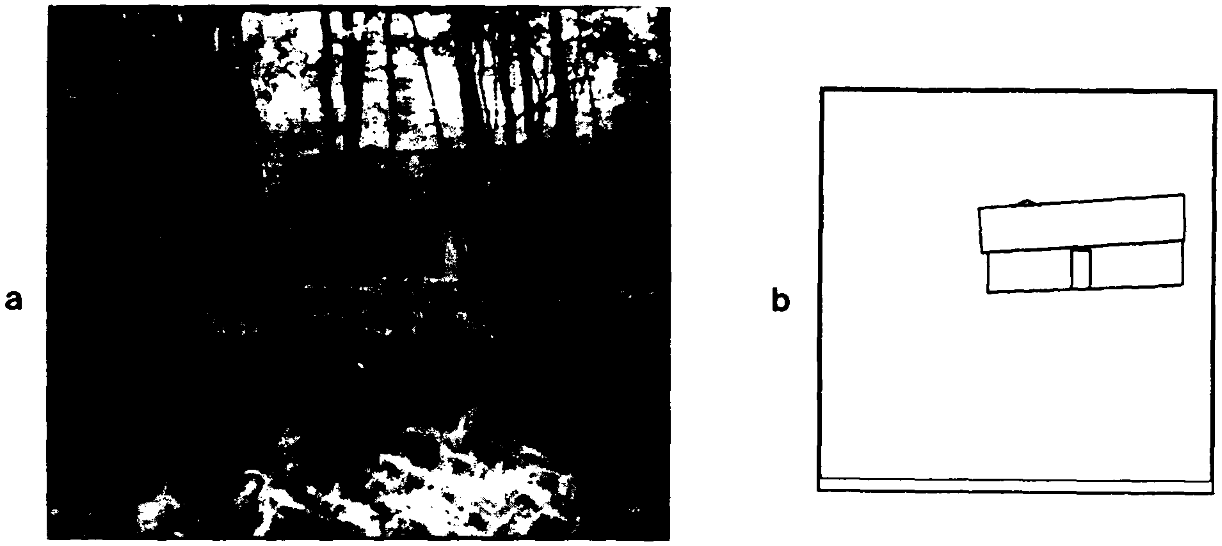


Figure 9: a) A challenging house recognition scene. b) The projection of the house.

included. Finally, the best match in this restricted space was found using a first-improvement local search strategy on a $k = 1$ add/delete neighborhood. The data lines corresponding to the best matches on the first image are shown in Figure 8b. The proportionality constants of the match error function were set to $\alpha = 0.5$ and $\beta = 4$ for all the matches presented in this section.

In these experiments, the system was asked to find the six landmarks in each of the six images. In 26 of the 28 cases where the landmarks were visible the matching system found the correct matches. For landmarks not visible the match error for the matches found by the system were so high that they were readily discarded. The first mismatch returned by the system was the result of only starting local search using the single model position associated with the highest peak in the transformation space. Initiating a local search from the second highest peak in the transformation space led directly to the correct match, and the match error for the correct match was half that of the incorrect match. This error simply points to the need to initiate local search from several distinct initial matches. However, the second mismatch indicates a more serious source of potential problems. When the structure of a landmark is not distinctive, such as is the case for two parallel vertical lines, there is a chance that the 'wrong' match will be found. This happened for the telephone pole on the right, which was matched to the data line segments associated with the street lamp. To correct this error we merged the righthand telephone pole and the street lamp into a single model, and the resulting hybrid model was unambiguous and successfully matched. However, the general problems posed by ambiguous models will demand more attention and study.

Note that several of the landmarks shown contain underconstrained 2D models where the optimal translation is not fully determined: the lefthand telephone pole, the street lamp and the side of the walkway. In these cases all model line segments are parallel, and the matrix M , defined in appendix A, is singular. The optimal translation, which is computed using M^{-1}, is therefore undefined. Since models of this type are important, the fitting scheme must be modified to accommodate them. In these results an additional term was added to the spatial fit measure. This term minimized distance, projected onto the model line, between data line segment end-points and the mid-point of the model line. Problems arise for this approach when model lines are near, but not exactly parallel. The proper solution is to check for ill-conditioning of M and independently to solve for the model's lateral and longitudinal placement (along the axis of the model lines). For a more detailed discussion of these issues see [BWR89].

The last result we present demonstrates the power of our 2D matching system to overcome serious weaknesses in the bottom-up line data. The 512x512 house scene image, Figure 9a, has been part of our house scene library for over ten years. It has always represented an extremely difficult challenge for natural scene interpretation. The bottom-up line data for this image is shown in Figure 10a. Note the tremendous amount of fragmentation is primarily a consequence of ambiguity caused by the tree texture and shadows, not any inherent weakness of the line extraction algorithm. A projection of the main portion of the house was specified by hand, including only the roof lines, walls, and door frame (Figure 9b). The matching system was then applied to find this model in the line data (Figure 10)a. Loose spatial constraints were assumed - 100 pixels and 0.3 radians of its true location. The peak in the transform space indicated the correct transformation, and the local search process, applied in the resulting

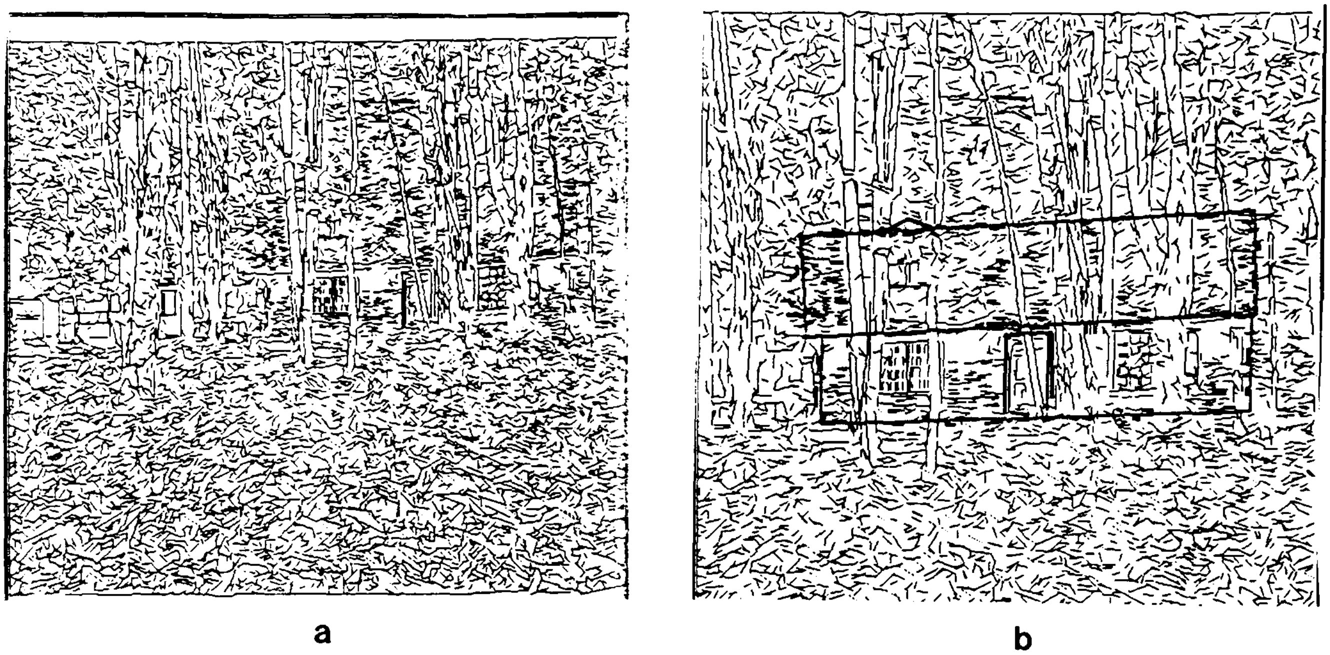


Figure 10: a) The bottom-up line data for the entire image. b) The optimal match to the house projection.

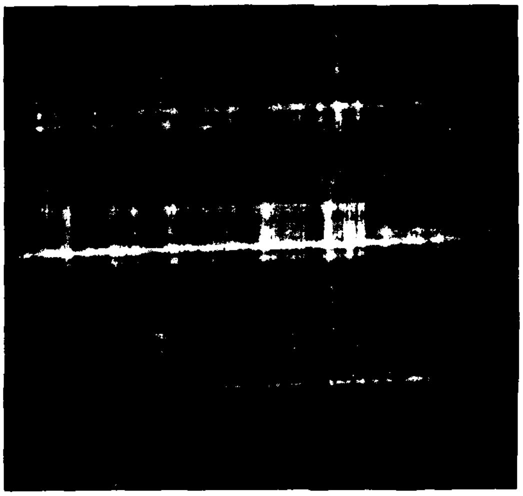


Figure 11: A slice of $\phi = 0$ through the transformation space. The bright spot in the center is the largest peak and is associated with the optimal transformation.

restricted correspondence space, produced the final match shown in Figure 10b. This is an exciting result.

Since the model was initially placed at the proper location, the peak in the transformation space should ideally be at u, v, and ϕ all equal to 0. A slice of constant $\phi = 0$ through the transformation space is shown as an intensity image in Figure 11. The bright spot at the center of the image is the peak in the space. Note that the peak lies on a long horizontal ridge corresponding to those transformations in which the model roof lines lie over the fragmented roof lines in the data. The vertical streaks are a consequence of the vertical model lines lying on top of vertical data line segments associated with the tree trunks. Although it was fortuitous that the highest peak in the transformation space corresponded to the optimal match, the optimal match would in general be found among the most significant peaks.

7. Conclusion

The problem of matching 2D models to imperfect line data has been formalized as a combinatorial optimization problem. At the heart of this formulation is a spatial fitting problem for registering model and data. We have presented a fit measure demonstrably superior to the reasonable alternatives for imperfect data. Moreover, we have presented a closed-form solution for determining the rotation and translation parameters that optimize this measure. We have applied local search techniques, derived for solving difficult combinatorial optimization problems, to the vision problem of 2D model matching Local search, combined with a means of generating reasonable initial matches, has proven to be a powerful matching tool. Finally, the combined results of this paper and [Kum89] demonstrate that 2D matching followed by 3D pose refinement is an effective means of solving a class of 3D identification problems, specifically the problem of updating robot position through landmark recognition.

A version of the matching system is being written in C for use on the Sequent Multi-processor. The C system will be used for real-time or near real-time robot navigation in our outdoor campus domain.

The study of alternative objective functions for fitting the data lines to the model is interesting in its own right, and a more complete discussion will be found in [BWR89]. In addition, the generalized hough transform and the case of models consisting only of parallel lines will receive more attention in the Tech Report.

Acknowledgements

We wish to acknowledge the insightful contributions of Peter Giblin and John Oliensis. In addition we appreciate the ongoing support of 'Teddy' Kumar, Brian Burns, Robert Collins, Al Hanson and Bruce Draper, along with the other members of the UMass VISIONS Lab.

A. Closed Form Solution Minimizing Model Projection Lateral Displacement

As presented in Section 4. the error function to be minimized is written as:

$$E_{\text{fit}} = \sum_i \ell_i \left(n_i \cdot (R(p_i) - T) - \rho_i \right)^2 \tag{3}$$

$$R = \begin{vmatrix} \cos\phi & -\sin\phi \\ \sin\phi & \cos\phi \end{vmatrix} \qquad T = \begin{vmatrix} u \\ v \end{vmatrix} \tag{4}$$

E_{fit} is the weighted sum of squared perpendicular distances between data line end-points $p_i = (x_i, y_i)$ and corresponding model lines l_i. Each model line, l_i has unit normal n_i, length ℓ_i, and its distance from the origin in the direction n_i is ρ_i. Differentiating with respect to T yields:

$$\frac{dE_{\text{fit}}}{dT} = 2 \sum_i \ell_i \left(n_i \cdot (R(p_i) - T) - \rho_i \right) n_i \tag{5}$$

The following substitution is helpful:

$$R(p_i) = S_i \hat{R} \qquad \text{where} \qquad S_i = \begin{vmatrix} x_i & -y_i \\ y_i & x_i \end{vmatrix} \qquad \hat{R} = \begin{vmatrix} \cos\phi \\ \sin\phi \end{vmatrix} \tag{6}$$

Setting the first partial of E_{fit} with respect to T equal to zero, and solving for the optimal T given ϕ yields:

$$\sum_i \ell_i \left(n_i \cdot T\right) n_i = \sum_i \ell_i \left(n_i \cdot S_i \hat{R}\right) n_i - \ell_i \rho_i n_i \tag{7}$$

Observe that

$$\left(n_i \cdot T\right) n_i = n_i n_i^T T \qquad \text{and} \qquad \left(n_i \cdot S_i \hat{R}\right) n_i = n_i n_i^T S_i \hat{R} \tag{8}$$

Finally

$$T = B\hat{R} - V \tag{9}$$

Where B and V are defined as follows:

$$M = \sum_i \ell_i n_i n_i^T \qquad B = M^{-1} \sum_i \ell_i n_i n_i^T S_i \qquad V = M^{-1} \sum_i \ell_i \rho_i n_i \tag{10}$$

The matrix M is singular or nearly singular, *iff* the lines in the model are parallel or nearly parallel. This case, as noted in the body of the paper, requires an additional constraint to define the optimal translation.

Substituting T in terms of $\hat{R}$ from equation 9 back into equation 4, we obtain E', the fit at the optimal translation for a specified rotation ϕ.

$$E' = \sum_i \ell_i \left(n_i \cdot \left(W_i \hat{R} + V\right) - \rho_k\right)^2 \qquad \text{where} \qquad W_i = S_i - B \tag{11}$$

The optimal rotation is a root of the first derivative of E' with respect to ϕ.

$$\frac{dE'}{d\phi} = c_1 \cos^2\phi - c_1 \sin^2\phi + c_2 \cos\phi \sin\phi + c_3 \cos\phi + c_4 \sin\phi \tag{12}$$

$$\begin{aligned}
c_1 &= \sum_i \ell_i (n_i \cdot W_i^1)(n_i \cdot W_i^2) \\
c_2 &= \sum_{kij} \ell_i \left((n_i \cdot W_i^2)^2 - (n_i \cdot W_i^1)^2\right)
\end{aligned}$$

$$\begin{aligned}
c_3 &= \sum_{kij} \ell_i \left((n_i \cdot V) - \rho_k\right)(n_i \cdot W_i^2) \\
c_4 &= -\sum_{kij} \ell_i \left((n_i \cdot V) - \rho_k\right)(n_i \cdot W_i^1)
\end{aligned}$$

W_i^κ is a vector formed from the column κ of matrix W_i.

Shifting terms involving $\sin\phi$ to one side of equation 13, squaring both sides, and substitute $\cos^2\phi - 1$ for $\sin^2\phi$ yields a fourth order equation in $\cos\phi$.

$$\cos^4\phi + a_1 \cos^3\phi + a_2 \cos^2\phi + a_3 \cos\phi + a_4 = 0 \tag{13}$$

$$\begin{aligned}
a_1 &= \frac{4c_1c_3 + 2c_2c_4}{4c_1^2 + c_2^2} & \qquad a_3 &= \frac{-2(c_1c_3 + c_2c_4)}{4c_1^2 + c_2^2} \\
a_2 &= \frac{c_3^2 - 4c_1^2 + c_4^2 - c_2^2}{4c_1^2 + c_2^2} & \qquad a_4 &= \frac{c_1^2 - c_4^2}{4c_1^2 + c_2^2}
\end{aligned}$$

The roots of equation 14 are found analytically and the optimal rotation ϕ^* must correspond to one of these four roots. The optimal translation follows directly from equation 9.

REFERENCES

[Bal81] D. H. Ballard. Generalizing the hough transform to detect arbitrary shapes. *Pattern Recognition*, 13(2):111 - 122, 1981.

[BS83] D. H. Ballard and D. Sabbah. "Viewer independent shape recognition", *IEEE Trans. Patt. Anal. Mach. Intel.*, PAMI-5(6):653-660, Nov. 1983.

[BC82] R. C. Bolles and R. A. Cain. "Recognizing and locating partially visible objects: the local-feature-focus method", *International Journal of Robotics Research*, 1(3):57-82, 1982.

[Bro81] R. Brooks. "Symbolic Reasoning Among 3-D Models and 2-D Images", Artificial Intelligence, 17: 285-348, 1982.

[Bro82] R. A. Brooks (1982), "Model-based three-dimensional interpretations of two-dimensional images", *IEEE Trans. Patt. Anal. Mach. Intel.*, PAMI-5(2):140-150, March 1982.

[BHR86] J. B. Burns, A. R. Hanson, and E. M. Riseman. Extracting straight lines. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, PAMI-8(4):425 - 456, July 1986.

[BWR89] J. Ross Beveridge, Richard Weiss, and Edward M. Riseman Matching models to broken and skewed line data. Technical Report (In Preparation), University of Massachusetts, Amherst, 1989.

[FHR89] C. Fennema, A. Hanson, and E. Riseman, "Towards Autonomous Mobile Robot Navigation", *Proc. DARPA Image Understanding Workshop*, Morgan Kaufman Publishers, Palo Alto, CA, May 1989.

[Goa83] C. Goad. "Special purpose automatic programming for 3D model-based vision", *Proc. DARPA Image Understanding Workshop*, Arlington, VA, 1983, 94-104, 1983.

[GL84] W.E.L. Grimson and T. Lozano-Perez. "Model-Based Recognition and Localization from Sparse Range or Tactile Data", *International Journal of Robotics Research*, 3(3):3-35, 1984.

[Hor87] B. K. P. Horn. Closed-form solution of abolute orientation using unit quarternions. *Journal of the Optical Society of America*, 4:629 - 642, 1987.

[HU87] D.P. Huttenlocker and S. Ullman. Object recognition using aligment. *First Int. Conf. on Computer Vision*, 102 111, June 1987.

[KSS86] A. Kalvin, E. Schonberg, J.T. Schwartz, and M. Sharir. Two-dimensional, model-based, boundary matching using footprints. *International Journal of Robotics Research*, 5(4):38 - 55, 1986.

[Kum89] R. Kumar. Determination of camera location and orientation. In *Proceedings: Image Understanding Workshop*, page (these Proceedings), Los Altos, CA, June 1989. DARPA, Morgan Kaufmann Publishers, Inc.

[LW88] Y. Lamdan and H.J. Wolfson. Geometric hashing: a general and efficient model-based recognition scheme. *Proc IEEE Second Int. Conf. on Computer Vision*, pages 238 - 249, Tampa, Dec. 1988.

[LHF88] Y. Liu, T. S. Huang, and O. D. Faugeras. Determination of camera location from 2d and 3d line and point correspondences. In *Proc. CVPR 88*, pages 82 88, Ann Arbor, June 1988.

[LK73] S. Lin and B. Kernighan. An effective heuristic algorithm for the traveling salesman problem. *Operations Research*, 21:498 - 516, 1973.

[Low80] D. G. Lowe. Solving for the parameters of object models from image desciptions. In *Proc. ARPA Image Understading Workshop*, pages 121 - 127, 1980.

[Low85] D. G. Lowe. *Perceptual Organization and Visual Recognition*. Kluwer Academic Publishers, 1985.

[PS82] C. H. Papadimitriou and K. Steiglitz *Combinatorial Optimization: Algorithms and Complexity*, chapter Local Search, pages 454 - 480. Prentice-Hall, Englewood Cliffs, NJ, 1982.

[SS85] J.T. Schwartz and M. Sharir. Identification of partially obscured objects in two and three dimensions by matching of noisy *characteristic curves*. Robotics Report # 46, Robotics Activity, New York University, 1985.

[SHD84] T. M. Silberberg, D. Harwood and L. S. Davis. "Object recognition using oriented model points". Tech Report CS-TR-1387, Center for Automation Research, Univ. of Maryland, College Park, MD 20740, 1984.

[TM87] D. Thompson and J. Mundy. "Three-Dimensional Model Matching from an Unconstrained Viewpoint", *Proceedings IEEE International Conference on Robotics and Automation*, 208 220, 1987.

[TFF88] L. W. Tucker, C. R. Feynman, and D. M. Fritzsche. Object recognition using the connection machine. In *Proc. CVPR 88*, pages 871 878, Ann Arbor, June 1988.

DEVELOPING THE ASPECT GRAPH REPRESENTATION FOR USE IN IMAGE UNDERSTANDING [1]

Kevin Bowyer, David Eggert, John Stewman and Louise Stark

Department of Computer Science and Engineering
University of South Florida
Tampa, Florida 33620
(813) 974-3032
kwb@usf.edu

Abstract.

This report gives an overview of several recent results in the development of the aspect graph representation for use in image understanding. First, a simple 3-D object recognition system has been implemented which uses the aspect graph as its basic object representation. Experiments with this system show that the aspect graph offers some important advantages for recognition systems in which an optimization technique is used to estimate parameters of translation and orientation. Second, an algorithm is described for creating the perspective projection aspect graph of general polyhedral objects. An implementation of this algorithm is in progress which will be able to take its input description of objects from the PADL-2 solid modeler. Third, an algorithm is described for creating the orthographic projection aspect graph of solids of revolution which have been defined as right, circular, straight homogeneous generalized cylinders. Other efforts in progress are aimed at developing algorithms to create the aspect graph of a larger class of curved objects and implementing a more robust object recognition system for general polyhedral objects.

1 Introduction

The introduction of the *aspect graph* concept is generally credited to Koenderink and van Doorn [16], who described a graph structure which they actually referred to as the *visual potential* of an object. The commonly agreed upon elements of the definition of an aspect graph are that (1) a node represents a 'stable view' of the object as seen from some maximal connected region of viewpoint space, (2) an arc represents a possible transition between such stable views by crossing from one such region of viewpoint space to a neighboring region, (3) there is a node for each such stable view, and (4) there is an arc for each such transition between such views. Figure 1 depicts an example of the aspect graph for a tetrahedron.

The aspect graph concept seems, intuitively, to be very powerful for use in image understanding and has recently become a popular topic of research. A large and growing number of researchers have discussed algorithms for generating some flavor of aspect graph representation and/or possible uses for it. Figure 2 depicts a categorization of the algorithms developed to date for automatically generating the aspect graph. Perhaps the most fundamental distinction between the different representations is whether viewpoint space is modeled as the Gaussian sphere, implying an *orthographic projection aspect graph*, or as all of 3-D space, implying a *perspective projection aspect graph*. A primary advantage of the Gaussian sphere model is that it allows an approximate aspect graph to be generated by first creating a uniform tessellation of the sphere and then grouping facets of the tessellation which see the same view of the object. An algorithm which operates along these lines can, in principle, be applied to objects of any geometric definition. Several researchers have used this form of aspect graph as the initial object description from which an automated recognition strategy is derived (*interpretation tree* [13], *prediction hierarchy* [1], *strategy tree* [9], *decision tree* [24], ...). Researchers have also developed several different algorithms to create the (orthographic or perspective projection) aspect graph of general polyhedral objects based on an exact partition of viewpoint space derived from the geometric definition of the object. Relatively

[1] This work was previously supported by Air Force Office of Scientific Research grant AFOSR-87-0316, and is currently supported by Air Force Office of Scientific Research grant AFOSR-[illegible] and National Science Foundation grant IRI-8817776.

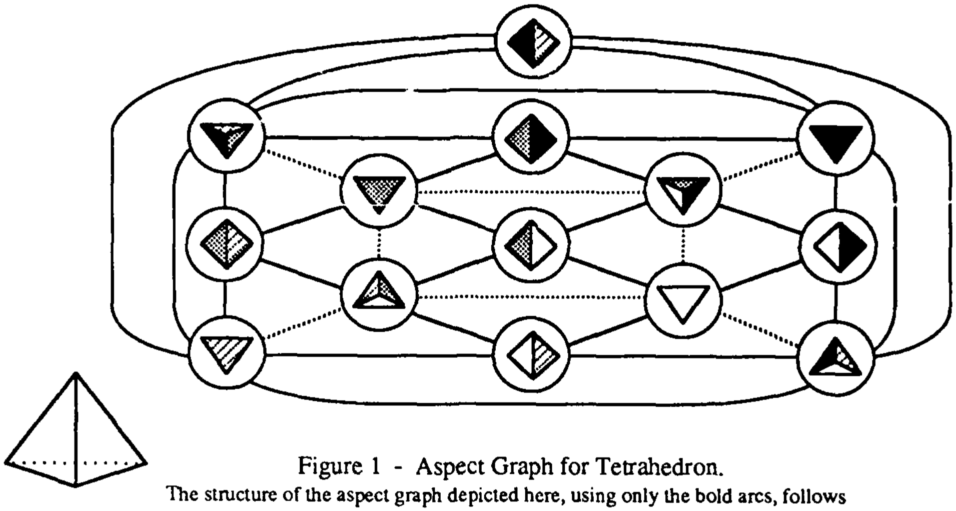


Figure 1 - Aspect Graph for Tetrahedron.
The structure of the aspect graph depicted here, using only the bold arcs, follows the convention of [Koenderink and van Doorn, 1979]. Addition of the dotted arcs would modify the graph to follow the convention of [Plantinga and Dyer, 1986].

less work has been done in applying this flavor of aspect graph in object recognition systems, in part because implementation of such an algorithm is a non-trivial task in and of itself.

Section 2 of this paper describes a simple recognition system which uses the perspective projection aspect graph as its basic object representation. This system illustrates some of the advantages of using the aspect graph for object recognition. Section 3 outlines an algorithm for creating the perspective projection aspect graph of general polyhedral objects. This algorithm is being implemented (in C on a SUN workstation) and an interface has been developed to read PADL-2 object definition files. The implementation will be made available to interested research groups. Section 4 describes an algorithm to create the orthographic projection aspect graph of solids of revolution. To our knowledge, this is the first algorithm to automatically create the exact aspect graph for any class of curved objects. We also believe this algorithm illustrates a general conceptual approach which can be applied to other classes of curved objects. Section 5 discusses certain problems with using the aspect graph for object recognition and suggests some lines of future research.

2 A Simple Recognition System Using Aspect Graphs

Several researchers have described approaches to object recognition which use an iterative technique to recognize an object from a library of models and estimate the parameters of translation and orientation [12, 17, 32]. The major problems encountered are (1) how to choose starting parameter estimates, and (2) how to know when the global minimum has been found. Since the aspect graph is based on a parcellation of viewpoint space into cells from which fundamentally different views are seen, it seems that it should provide a solution to these problems. In order to investigate empirically whether this is true, we implemented a simple recognition system using our early algorithm to create the aspect graph of convex polyhedra. There are three basic modules to the system [27].

One module takes the boundary surface description of a convex polyhedral object as its input and creates the perspective projection aspect graph. The important elements of the aspect graph representation for our purposes are that each node is attributed with (1) a definition of the corresponding 3-D cell of viewing space, (2) a definition of the object faces visible from that cell, and (3) the coordinates of a 'central viewpoint' in the cell. The individual aspect graphs of all the models in the database are merged into an *equivalence class graph* which distinctly represents only the aspect- and/or cell-equivalent classes of views [30]. Two nodes are *aspect equivalent* iff the faces visible in the aspect for one node can be exactly mapped onto the visible faces for the other node by a suitable change in scale and orientation. Two nodes are *cell equivalent* iff (1) they are aspect equivalent, and (2) the cell of viewing space for one node can be exactly mapped onto the cell of viewing space for the other by the same change in scale and orientation used for aspect equivalence. Further, the equivalence class graph is arranged by levels, where all nodes in a given level have the same number of visible faces.

A second module implements a Fourier Descriptor (FD) based feature extraction and matching strategy. First, a line drawing image is converted to a graph description where arcs correspond to edges and nodes correspond to vertices. Then a unique subset of circuits in this graph, representing the face outlines, is selected. The center

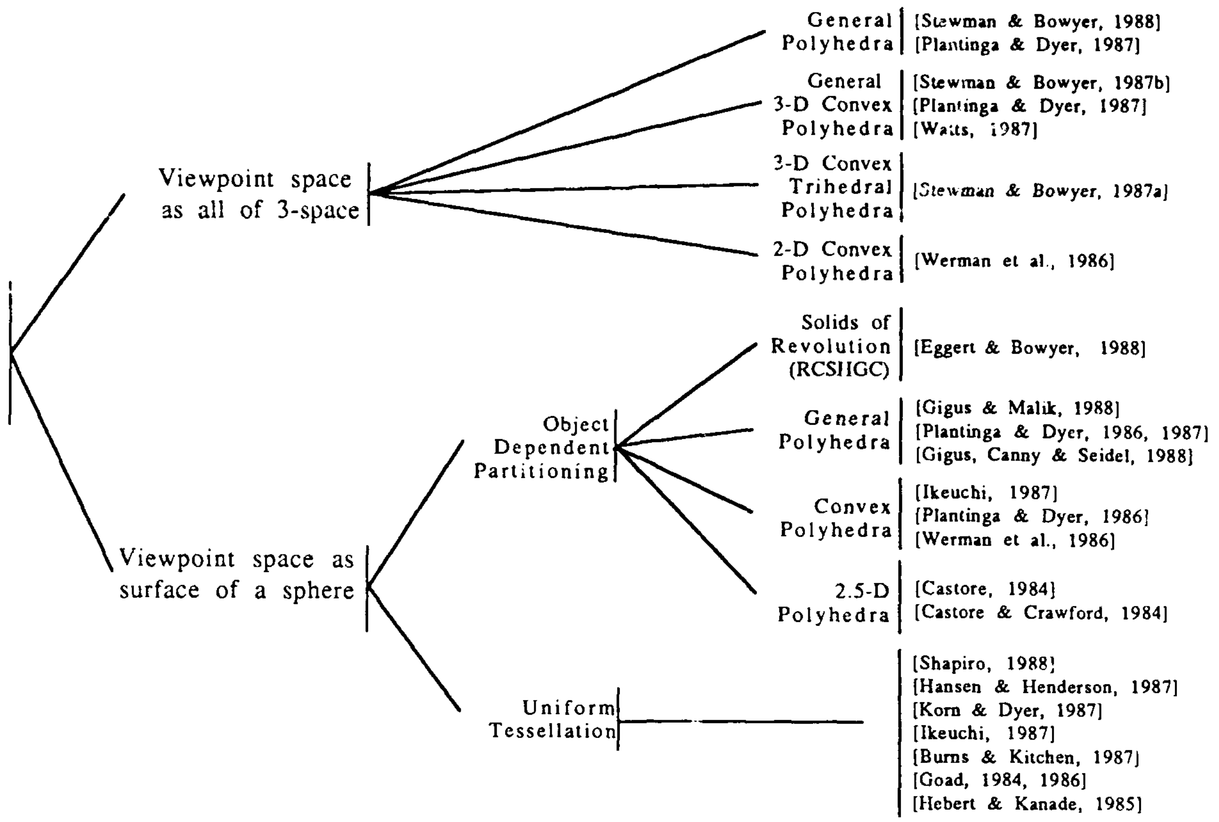
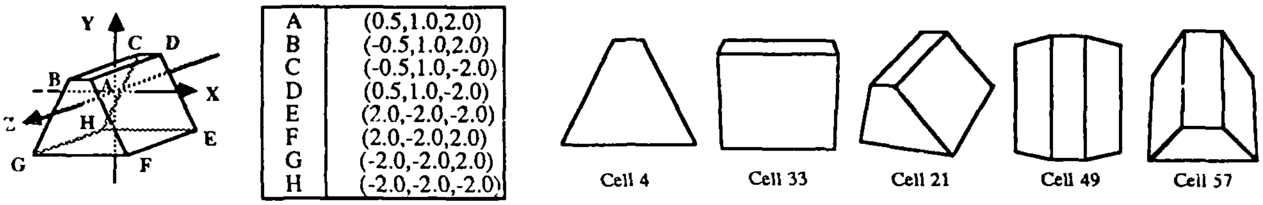


Figure 2 - Algorithms for Aspect Graph Creation.
References which describe an *algorithm* for automatically creating the aspect graph for any object in some defined class are categorized by model of viewpoint space and class of object geometry.

of mass of the centers of these circuits is found, and a pattern of rays from the center of mass to the individual circuit centers is created. An analogous ray pattern for a model is rotated, translated and scaled to best fit the pattern from the image. The figure of merit for the match is the sum of the squares differences of the FDs for each of the model-object circuit pairs. Thus, the feature matching reports a figure of merit for the match, along with the 2-D rotation, 2-D translation, and scale used in the best fit of model projection to observed line drawing. While this particular strategy may not extend to a robust, practical recognition system, it does allow us to explore tradeoffs which are representative of a broader class of possible strategies. In principle, any feature matching strategy which provides a suitable figure of merit could be used.

The third module uses an iterative technique to search the database and recognize the object. Since the equivalence class graph is arranged by levels, only aspects whose number of visible faces matches the number of circuits found in the line drawing are considered as candidates. For each such node, an iterative technique (damped least squares) is used to find the best-match viewpoint within the corresponding cell. The search for a match involves a six-parameter space: three parameters of translation, $[X, Y, Z]$, and three parameters of rotation, $[R_x, R_y, R_z]$. The feature matching module provides information on the change in scale, rotation, and translation used to give the best match of circuits from the model projection to circuits from the image. These four values have a loose correspondence to four of the six parameters in the search: the 2-D scale change to Z, the 2-D rotation to R_z, and the 2-D translation to X and Y. Using these values in the search process allows the optimization algorithm to search a two-parameter space (R_x and R_y). One iteration in the search process consists of making a 2-D feature match, immediately updating X, Y, Z, and R_z based on the results of the 2-D feature match, and using damped least squares to find new values for the remaining parameters.

The basic idea behind using aspect graphs to guide recognition is fairly simple. Assume that we have a database in which each object is represented by its (perspective projection) aspect graph, and that we are given an image in which an unknown object from the database appears at an unknown orientation and translation. We generate a separate solution for each node by picking a starting point inside each viewing cell and constraining the iterative technique to find a solution within that cell. This addresses the first problem with using an iterative technique for recognition, that of how to choose starting parameter estimates. We then select the best solution



A	(0.5,1.0,2.0)
B	(-0.5,1.0,2.0)
C	(-0.5,1.0,-2.0)
D	(0.5,1.0,-2.0)
E	(2.0,-2.0,-2.0)
F	(2.0,-2.0,2.0)
G	(-2.0,-2.0,2.0)
H	(-2.0,-2.0,-2.0)

(a) Truncated wedge and vertex coordinates

(b) Initial views of 1, 2, 3, and 4 face aspects

Figure 3 - Truncated wedge model with representative one, two, three and four-face aspects.

Table 1 - Summary of viewpoint percent error (VP) for results of matching to simulated views.

Each cell was tested with 100 randomly generated simulated views. VP is calculated as the ratio of the distance from the recognized viewpoint to the correct viewpoint, divided by the distance from the correct viewpoint to the origin. Thus VP is a measure of the aggregate error in the three parameters of translation and three parameters of rotation.

Aspects	1E-6 ≤ VP < 1E-4	1E-4 ≤ VP < 1E-2	0.01 ≤ VP < 1.0
1 face view (cell 4)	6	92	2
2 face view (cell 33)	5	87	8
3 face view (cell 21)	88	12	0
3 face view (cell 49)	76	19	5
4 face view (cell 57)	94	6	0

across all nodes as the recognized view of the object. This addresses the second problem with using an iterative technique, that of knowing when the global minimum has been found.

The validity of this approach rests on two assumptions. First it must be the case that, for the viewing cell which does contain the correct viewpoint, the search process for that cell converges to the correct point. Second, it must be the case that the solution found in the correct viewing cell has a better figure of merit than the solution found in any other cell.

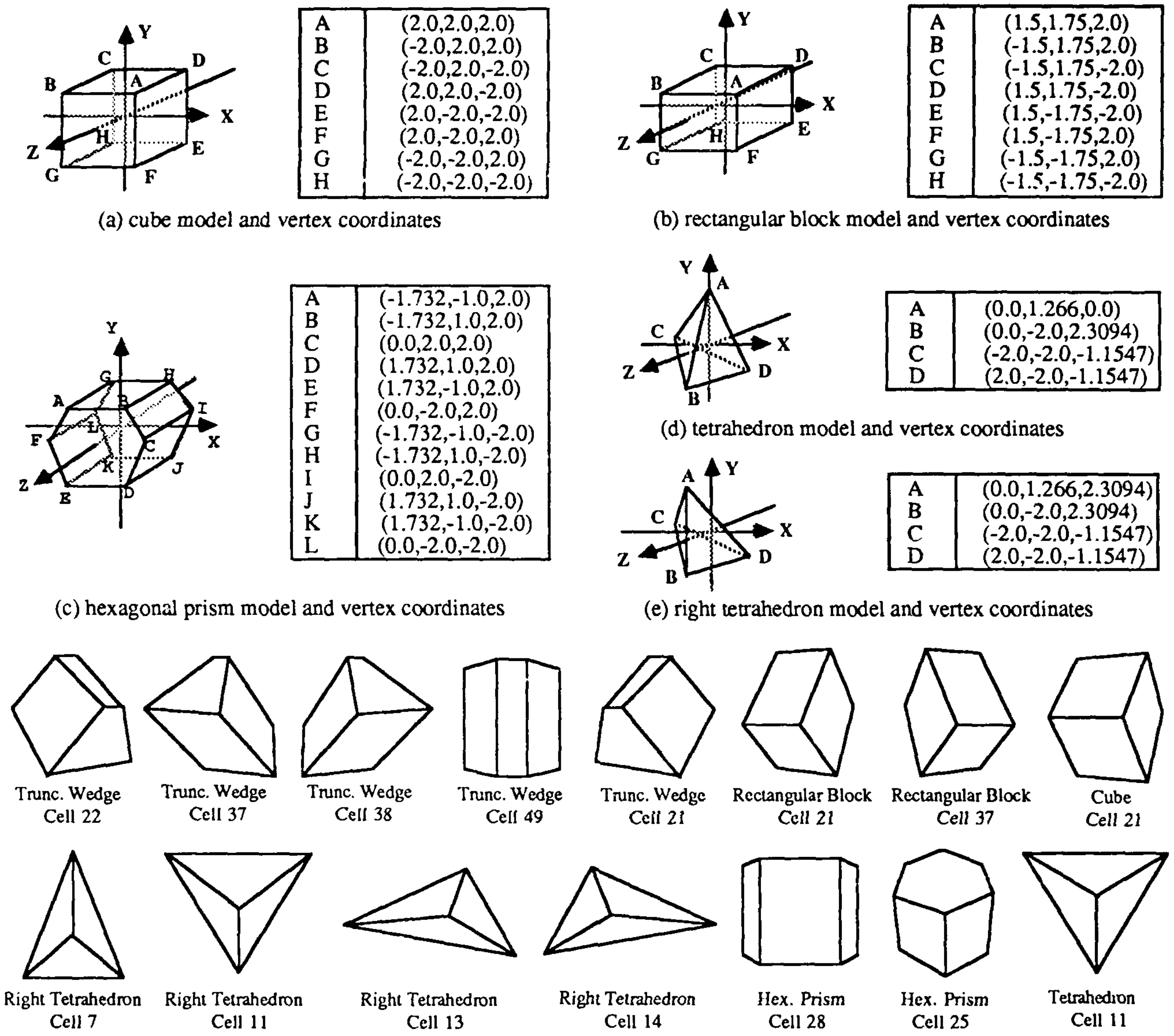
2.1 Testing the Validity of the Approach

In order to assess the validity of using the aspect graph to guide the search process for recognition, a series of simulated recognition experiments was carried out. One set of recognition trials was aimed at assessing how well the search process converges to the correct viewpoint parameters v[illegible]e correct viewpoint is, in fact, in the viewing cell under consideration. Another set of trials was ai[illegible]ssessing how well the approach does at recognizing the correct object by selecting the lowest objective function value across a set of candidate aspects.

2.1.1 Reaching the Correct Solution Within a Cell

For the first set of recognition experiments, we randomly generated 100 simulated views within representative one-, two-, three-, and four-face aspects of a truncated wedge, as illustrated in Figure 3. The simulated views were generated as follows. First, assuming a maximum viewer-to-object distance, points were randomly generated within a sphere centered around the object and the first 100 points falling inside each cell were kept. Each point generated in this way specifies three of the viewing parameters: R_x, R_y, and Z. Additionally, for each point, the rotation around the line of sight, R_z, was varied randomly between 0 and 360 degrees. Also, a random offset in the range of plus or minus 5 percent of the distance from the simulated viewpoint to the origin was selected for each of the X and Y parameters. The viewing parameters generated in this way were used to create 100 simulated views for each of the five aspects. For each of the 100 simulated views of each of the five selected aspects, the search procedure was begun with standard initial parameter estimates for that cell.

Table 1 summarizes the results of this first experiment. None of the 500 trials (100 trials each for 5 cells) converged to a viewpoint estimate that was off by more than one half of one percent of the distance from the correct viewpoint to the origin, and the vast majority were substantially closer. We believe this data strongly



A	(2.0,2.0,2.0)
B	(-2.0,2.0,2.0)
C	(-2.0,2.0,-2.0)
D	(2.0,2.0,-2.0)
E	(2.0,-2.0,-2.0)
F	(2.0,-2.0,2.0)
G	(-2.0,-2.0,2.0)
H	(-2.0,-2.0,-2.0)

(a) cube model and vertex coordinates

A	(1.5,1.75,2.0)
B	(-1.5,1.75,2.0)
C	(-1.5,1.75,-2.0)
D	(1.5,1.75,-2.0)
E	(1.5,-1.75,-2.0)
F	(1.5,-1.75,2.0)
G	(-1.5,-1.75,2.0)
H	(-1.5,-1.75,-2.0)

(b) rectangular block model and vertex coordinates

A	(-1.732,-1.0,2.0)
B	(-1.732,1.0,2.0)
C	(0.0,2.0,2.0)
D	(1.732,1.0,2.0)
E	(1.732,-1.0,2.0)
F	(0.0,-2.0,2.0)
G	(-1.732,-1.0,-2.0)
H	(-1.732,1.0,-2.0)
I	(0.0,2.0,-2.0)
J	(1.732,1.0,-2.0)
K	(1.732,-1.0,-2.0)
L	(0.0,-2.0,-2.0)

(c) hexagonal prism model and vertex coordinates

A	(0.0,1.266,0.0)
B	(0.0,-2.0,2.3094)
C	(-2.0,-2.0,-1.1547)
D	(2.0,-2.0,-1.1547)

(d) tetrahedron model and vertex coordinates

A	(0.0,1.266,2.3094)
B	(0.0,-2.0,2.3094)
C	(-2.0,-2.0,-1.1547)
D	(2.0,-2.0,-1.1547)

(e) right tetrahedron model and vertex coordinates

(f) set of 15 3-face aspects used in tests

Figure 4 - Additional object models used in tests along with 15 representative 3-face aspects.

supports the assumption that the search process will converge to the correct viewing parameters when started in the correct viewing cell, suggesting that the aspect graph can be used to enumerate a complete set of starting points for a search process.

2.1.2 Recognition by a Minimum Across Cells

For this experiment, we generated (in the same manner as described aoove) 25 random simulated views for each of 15 three-face aspects taken from the aspect graphs of six different objects. These 15 aspects ıepresent all of the three-face equivalence classes for the six objects. The additional object models and views are depicted in Figure 4. For each of the 25 simulated views in each of the 15 aspects, we executed the search process for each of the 15 aspects. Thus, for each of 375 simulated views, we have one recognition solution found in the correct viewing cell and fourteen solutions found in different incorrect viewing cells. We are interested in how often the result from the correct cell has a better figure of merit than that from any of the incorrect cells. Without exception for each of the 375 simulated views, the best figure of merit was the one found in the correct cell. Further, the best figure of merit from any of the incorrect cells was always several orders of magnitude different from that found in the correct cell (see Table 2). Thus the assumption that the figure of merit can be compared across cells also seems

Table 2 - Comparison of error values for correct and incorrect matches of simulated views to model aspects.

Object and Aspect in the simulated view	Largest error of 25 matches to correct model aspect	Smallest error of 350 matches to incorrect model aspects	Incorrect model aspect which resulted in smallest error
Truncated Wedge (Cell 21)	5.11E-11	4.51E-1	Rectangular Block (Cell 21)
Truncated Wedge (Cell 22)	1.11E-10	8.28E-1	Truncated Wedge (Cell 21)
Truncated Wedge (Cell 37)	4.97E-13	1.92E+0	Rectangular Block (Cell 37)
Truncated Wedge (Cell 38)	8.25E-11	3.57E-1	Truncated Wedge (Cell 22)
Truncated Wedge (Cell 49)	1.85E-9	1.75E+3	Hexagonal Prism (Cell 28)
Cube (Cell 21)	3.06E-14	1.56E+0	Rectangular Block (Cell 21)
Rectangular Block (Cell 21)	2.07E-13	1.06E-1	Truncated Wedge (Cell 21)
Rectangular Block (Cell 37)	7.27E-12	4.43E-1	Rectangular Block (Cell 21)
Tetrahedron (Cell 11)	1.29E-13	1.16E-3	Right Tetrahedron (Cell 11)
Right Tetrahedron (Cell 7)	2.00E-11	5.55E-1	Right Tetrahedron (Cell 11)
Right Tetrahedron (Cell 11)	1.45E-11	1.31E-3	Tetrahedron (Cell 11)
Right Tetrahedron (Cell 13)	1.77E-12	5.28E+1	Right Tetrahedron (Cell 7)
Right Tetrahedron (Cell 14)	2.19E-11	2.48E+1	Right Tetrahedron (Cell 7)
Hexagonal Prism (Cell 25)	2.19E-11	5.87E+2	Truncated Wedge (Cell 22)
Hexagonal Prism (Cell 28)	2.56E-1	2.54E+4	Truncated Wedge (Cell 49)

strongly supported by empirical data.

2.2 Discussion

The results of our simulated recognition experiments show that the perspective projection aspect graph provides a rational means of choosing a set of starting parameter estimates for each fundamentally different view of each object. This approach is more reasonable than generating initial parameter estimates at uniform increments in some of the parameters [12], generating a succession of random initial parameter estimates [32], or choosing starting parameter estimates from manually identified quasi-invariant features [17].

The system implemented for our simulated recognition experiments has a limited domain of competence, and is intended as a demonstration of concept rather than as a practical system. We have run some experiments with real data [27], but most of our effort is going into development of a new prototype system which will handle general polyhedral objects and use a more practical and efficient feature matching strategy.

3 Creating the Aspect Graph of Polyhedral Objects

At least three different approaches can be used to construct the perspective projection aspect graph of polyhedral objects. Stewman and Bowyer [29] presented an algorithm which creates the perspective projection aspect graph of convex polyhedra by finding all the lines and points of intersection between the planes in which the faces of the object lie and then enumerating all the nonempty cells in this parcellation of viewpoint space. Watts [33] presented another algorithm for this problem, based on using a "plane sweep" approach. Also, Edelsbrunner et al. [4] presented an algorithm for creating the *geometric incidence lattice* describing an *arrangement* of planes. For a convex polyhedral object, the geometric incidence lattice created for the planes in which the faces of the object lie correctly describes the parcellation of viewpoint space for the aspect graph. In this case, essentially all that is needed to derive the aspect graph from the incidence lattice is to distinguish one node as representing the object and to record the faces visible from each other node.

It is possible to extend any of these three approaches to handle general polyhedral objects. In view of the elegance and rigor of Edelsbrunner's work, we chose to use the geometric incidence lattice as the basis for developing an algorithm to create the perspective projection aspect graph of general polyhedral objects [31]. (Plantinga and Dyer [20] describe an algorithm somewhat similar to this, but using a different model of visibility.)

3.1 From Geometric Incidence Lattice to Aspect Graph

The *geometric incidence lattice* is a data structure which represents the *arrangement* of a set of hyperplanes in n-dimensional space. The geometric incidence lattice for a set of planes in 3-space has nodes representing *3-faces* (bounded subsets of 3-space), *2-faces* (bounded planar regions), *1-faces* (line segments or half-lines), and *0-faces* (points). Further, each k-face is linked to each of the (k-1)-faces which bound it, and to each of the (k+1) faces which it bounds.

The important differences between the geometric incidence lattice and the parcellation of space for an aspect graph are:

1. The planes in which the faces of a non-convex polyhedron lie are not sufficient to form the appropriate geometric incidence lattice. Additional surfaces arise due to self-occlusions. An ***auxiliary plane*** is introduced when a vertex visibly projects onto an edge which does not lie on the same face. For an object with N faces, there are $O(N^2)$ combinatorially possible edge-vertex pairs. An *auxiliary quadric surface* is introduced when the projection of three edges intersects at a point. There are $O(N^3)$ combinatorially possible edge triplets. To create an incidence lattice which can be used to derive the aspect graph, all of the object bounding planes, auxiliary planes and auxiliary quadric surfaces must be considered.

2. For nodes of the lattice which correspond to viewing cells, no record of what is visible in the associated view is inherent in the incidence lattice. Part of this information can be computed as the incidence lattice is constructed, but it seems that it can only be completely determined after the lattice is finished.

3. There is no distinction between object and viewpoint space in the geometric incidence lattice. In the lattice derived from the planes bounding a convex polyhedron, there is a single 3-face representing the interior of the object and all other 3-faces represent viewing cells. However, in the lattice for a non-convex polyhedron, there is a connected set of 3-faces which represents the object interior. Also, individual viewing cells may be represented by either a single 3-face or by a connected set of 3-faces.

Thus it is clear that the geometric incidence lattice and the aspect graph are related, but quite different, entities and that the use of the geometric incidence lattice to derive the aspect graph is not trivial. In principle, the required extensions and enhancements could be computed in a variety of different ways. However, in practice it becomes very important to compute things in an order such that the size of the intermediate data structures is kept to a minimum. Due to the possible existence of $O(N^3)$ auxiliary surfaces, the worst-case size of the incidence lattice is $O(N^9)$. Even if no auxiliary quadric surfaces are generated, the auxiliary planes may create a worst-case data structure of size $O(N^6)$. Our algorithm for the aspect graph of general polyhedral objects can be viewed as operating in four stages: (1) construction of the geometric incidence lattice for the planes in which the faces of the object lie, (2) updating this lattice with the necessary auxiliary planes, (3) determining the quadric surfaces involved in the lattice, and (4) finalizing the various attributes attached to the nodes of the aspect graph.

3.2 Constructing the Incidence Lattice for the Object Planes

The first stage in the algorithm is to construct the geometric incidence lattice for the arrangement of the planes in which the faces of the object lie. There are four steps to this stage: (1) listing the object planes, (2) constructing an initial spanning lattice, (3) adding the remainder of the object planes to the lattice, and (4) distinguishing object and viewpoint space in the lattice.

The first step, listing the planes associated with the faces of the object, is relatively simple. However, note that for non-convex polyhedra, (1) several faces may lie in the same plane, and (2) when this happens, a different subset of the faces may be visible to each side of the plane.

The second step is to construct an initial minimal lattice, $A(H_0)$, which spans 3-space. (See Figure 5.) Each k-face in the lattice is given an object/viewspace attribute which can take on one of four values: **boundary**, **interior**, **viewspace**, or **unknown**. All k-faces in $A(H_0)$ are initially marked **unknown**.

The third step is to update $A(H_0)$ to represent the arrangement defined by the complete set of object bounding planes, resulting in an expanded incidence lattice, $A(H_1)$ For each plane, **h**, not already in $A(H_0)$ perform the following three steps:

1. Locate a 1-face of the current lattice whose closure intersects **h**.

2. Using the 1-face found in the first step to enter the lattice, mark each face in the current lattice whose closure intersects **h**.

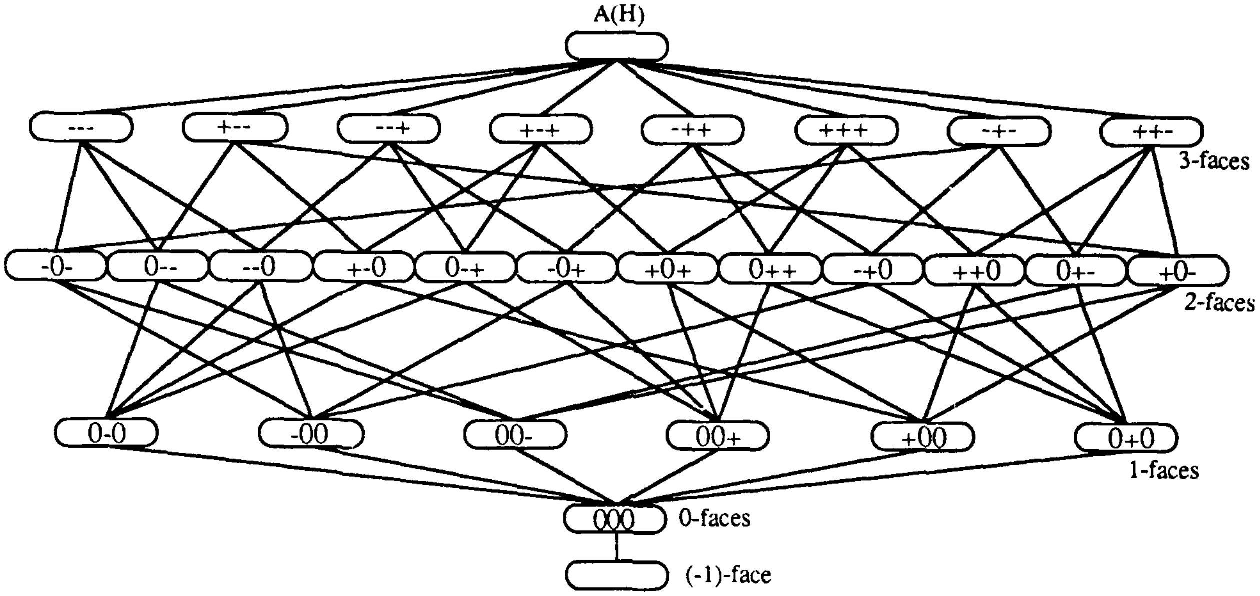


Figure 5 - Initial Incidence Lattice Spanning 3-Space – A(H_0).
As planes are added to the set H the incidence lattice will be updated to reflect the arrangement A(H) of the N planes in H. Each node is labeled with a face word w(f) = {d_1 d_2... d_N} such that each digit represents the relationship of that node to each of the N planes in H.

3. Update the marked faces in the lattice and integrate the new faces contained in **h** into the lattice.

With each new object bounding plane that is added to the lattice, some existing faces are split and some new ones are added. Each face that is split or added must have its object/viewspace attribute initialized to **unknown**.

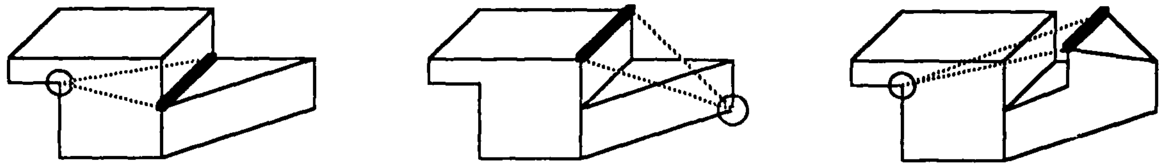
We also follow Edelsbrunner's notation in marking each node in the lattice, representing some k-face, *f*, with a *face word*, *w(f)*, within which each digit is -, 0 or +. These symbols represent the notions of inside, on, and outside, respectively. The face word is used to indicate the location of every k-face in the arrangement with respect to each of the planes. In addition, the closure of a k-face is the set of lower order faces whose face word differs from that of the k-face only where the lower order face word has a 0. In physical terms, the closure of a 3-face (volume) includes the surface patches, edges, and vertices that bound it. The closure of a surface patch includes the edges and vertices which form its border. The closure of an edge includes the vertices that are its end points.

The fourth step in this stage is to distinguish between object and viewpoint space. It is perhaps worth noting that *there is no common relationship between the various face words which describe the interior* 3-faces of a non-convex object. The process used to distinguish nodes representing object and viewpoint space is as follows:

1. Determine the boundary of the object in the lattice.
   The process is started by selecting a vertex, V_0, which is part of the original object description and using it to enter the lattice. At the completion of this process, the object/viewspace attributes of all nodes in the lattice are either **boundary** or **unknown**, so that the nodes which correspond to the boundary surface description of the object have been marked. Note that a non-convex object will have more face, edge and vertex nodes in the lattice than it does in the boundary surface description used to define the object.

2. *Determine the interior 3-faces of the object.*
   First locate a set of 3-faces which lie inside the object. This can be done by examining the 3-faces which lie to either side of one of the 2-faces on the boundary. Use these 3-faces with a process analogous to a seed fill algorithm to locate all other interior 3-faces, 2-faces, 1-faces, and 0-faces. Once the surface and interior of the object have been identified in this manner, the remaining nodes in the incidence lattice, those still marked **unknown**, comprise the viewing space. These can now be marked **viewspace**.

If the object is convex, then the lattice is correct and complete at the end of this stage of the algorithm. If the object is not convex, then there are auxiliary planes, and possibly auxiliary quadric surfaces, which must be determined and introduced into the lattice.



(a) e-v pair which passes edge shadow test but not convex hull test.

(b) e-v pair which passes convex hull test but not edge shadow test.

(c) e-v pair which passes both convex hull and edge shadow tests but not triangle test.

Figure 6 - e-v pairs which pass / fail different tests.

3.3 Introducing Auxiliary Planes Into the Lattice

A given edge, e, and vertex, v, generate an *active* auxiliary plane interaction iff a line can be drawn from v to some part of e without passing through the object. Thus the *active* auxiliary planes represent visible e-v interactions. For most objects, only a small fraction of the combinatorially possible e-v pairs generate active auxiliary planes. In principle, it is possible to either (1) form the incidence lattice using all potential auxiliary planes and then later group together cells which see equivalent views of the object, or (2) determine which of the potential auxiliary planes are active, and construct the incidence lattice using only these. In practice, the intermediate size of the lattice when all potential auxiliary planes are introduced makes it imperative to introduce only the active auxiliary planes. Therefore the second stage of the algorithm generates exactly the set of active auxiliary planes and introduces them into the lattice. The actual introduction of these planes into the lattice is done in the same way as for object planes, but a sequence of several different tests is used to efficiently generate exactly the set of active auxiliary planes.

An exact test for whether a given e-v pair generates an active visible interaction can be formulated as follows. Consider the triangle formed by e and v. This triangle represents all the possible lines of sight along which v projects on top of e. The test determines whether or not the object volume blocks all of the lines of sight from v to any point on e. The test is made by checking the e-v triangle against all the (convex) 3-faces of the object volume, as marked in the current lattice. If the object does block all the lines of sight, then the potential auxiliary plane is not active. If it does not, then the auxiliary plane is active and should be introduced into the lattice. We will call this the *triangle test.* Because the triangle test is relatively expensive to perform, we first use two cheaper, inexact tests to eliminate many e-v pairs from consideration. These tests are inexact in that they do not eliminate all e-v pairs which do not have a visible interaction.

One inexact test for active e-v pairs uses the convex hull of the object. Only those edges and vertices which lie on the boundary of faces which are not part of the convex hull of the object can possibly generate an active visual interaction. Thus, for "mostly convex" objects we can eliminate many of the e-v combinations by computing the convex hull of the object and only considering edges and vertices which are part of at least one face which is not on the convex hull.

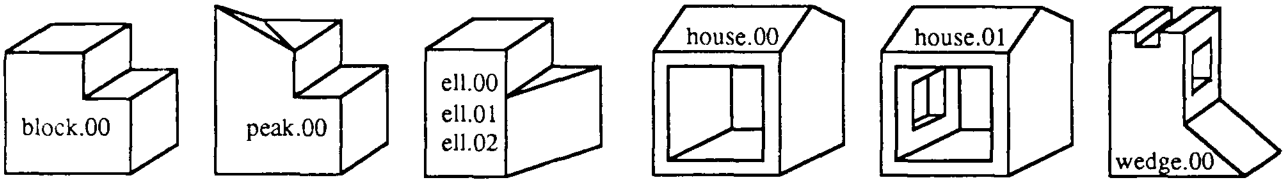
Another inexact test is, for a given e, consider a given v for possible interaction only if it does not lie in the *edge shadow* of v. The edge shadow test [6] checks the position of v with respect to the planes of the two faces which share e. The edge shadow is defined as the spatial volumes from which one or both of the faces sharing the edge cannot be seen. For this discussion, assume that a single face is visible from the 'front' side of the plane containing it and not visible from 'behind.' For a convex edge, the edge shadow is the single 'quadrant' which is 'behind' both of the planes. For a concave edge, the edge shadow is the three 'quadrants' which are behind one or both of the planes. If v lies in the edge shadow of e, then the auxiliary plane for this e-v pair cannot be active. Examples of e-v combinations which pass/fail these various tests are depicted in Figure 6. The effects of the various tests on cutting down the number of potential auxiliary planes for some sample objects are tabulated in Table 3. The first column gives the numbers of faces, edges, vertices and object planes for eight different nonconvex objects. The objects range from those having a single simple concavity to those having multiple through holes. The second column gives numbers of auxiliary planes that would be found if the object were treated as transparent. The third and fourth columns give the numbers of e-v pairs which pass the convex hull test or the edge shadow test, respectively. The fifth column shows the numbers which pass both tests together. The sixth column shows the results after use of the triangle test, representing exactly the number of active auxiliary planes.

3.4 Handling Quadric Surfaces

An auxiliary quadric surface is defined by the locus of viewpoints from which three non-coplanar edges project on top of each other. (Visual interaction of more than three edges occurs only at the intersection of surfaces

Table 3 - Effects of different tests on the number of auxiliary planes.

OBJECTS		Number of e-v pairs which pass test					TOTAL PLANES
Name	Description *	no test	convex hull	edge shadow	c.h. & e.s.	triangle	
block.00	[8{8},18,12]	21	5	5	5	5	13
peak.00	[9{9},19,12]	42	23	13	12	8	17
ell.00	[8{8},17,11]	28	3	6	3	3	11
ell.01	[11{11},23,15]	118	45	16	7	3	14
ell.02	[14{13},29,19]	200	86	36	21	3	16
house.00	[11{11},27,18]	99	6	22	6	6	17
house.01	[15{15},39,26]	207	52	92	48	40	55
wedge.00	[16{15},39,26]	153	123	82	74	29	44
* form of description is [faces{object planes},edges,vertices] on object surface							



# indicates how many holes in back of ell

defined by the interaction of three edges.) If the three edges are skew to one another, then the surface is a hyperboloid of one sheet. If the three edges are parallel to a common plane, then the surface is a hyperbolic paraboloid. While there are $O(N^3)$ combinatorially possible e-e-e triplets, most typical objects will have very few, if any, quadric surfaces in their parcellation of viewpoint space. The upper bound is only approached with very irregularly-shaped transparent objects or unusual polyhedral scenes. (See the example given by Plantinga and Dyer [20].)

Given that the incidence lattice has already been constructed for the object bounding planes and the active auxiliary planes. the set of possible edge triplets to consider for visual interaction defining an auxiliary quadric surface is greatly constrained. There are, in general, four planes created by the four edge-vertex pairs formed from two skew edges and their four vertices. These auxiliary planes bound two volumes of the viewing space which are of particular interest to us. These two volumes include cells from which the two edges appear to intersect. In one volume, one edge partially occludes the second. In the other volume, the second edge partially occludes the first. Outside of these two volumes neither edge occludes the other. The smallest sublattice containing each of these volumes can be identified, and marked as having a view where a partial occlusion of one edge by another exists.

An exact test for whether an e-e-e triplet generates an active quadric surface can be devised as follows. Given a pair of edges from different faces, such that neither edge lies completely in the edge shadow of the other, find the tetrahedron that represents the locus of all lines of sight from points on one edge to points on the other. In a fashion similar to the triangle test, the tetrahedron is tested against each of the **interior** 3-faces in the lattice to see if all lines of sight from one edge to the other are blocked. If so, then there is no visible intersection of the pair of edges, and so no e-e-e triplet containing that edge pair can generate an active quadric surface. A necessary condition for an e-e-e triplet to generate an active quadric surface is that all three pairs of edges pass this "tetrahedron test." For an e-e-e triplet where all three edge pairs visually interact, we then determine whether the volumes from which the interactions are visible have any mutual intersection. If so, then the e-e-e triplet generates an active quadric surface, whose relevance is restricted to the volume of mutual intersection. The quadric surface need only be introduced into the smallest sublattice containing this volume of mutual intersection.

3.5 Postprocessing to Finalize the Aspect Graph Structure

Once all of the object planes, auxiliary planes and auxiliary quadric surfaces have been added to the set of surfaces II, the arrangement of surfaces represents the underlying structure of both the object and the parcellation of the viewing space. Since the viewing space and object regions are attributed as such we can take the set of viewing space nodes (3-faces) and form from them cells in the aspect graph. To do so we must determine what the visible line drawing configuration is for each node. If the line drawing configuration does not change from one node to another then the two nodes are part of the same cell of viewing space for the aspect graph.

To determine which singularities are actually visible in the perspective projection of the object that is seen from a particular node in the lattice we use the following approach. We know for certain that no visual event occurs between any two viewpoints within the 3-face associated with a given node. In other words, a particular event is either visible from all viewpoints within the 3-face or is visible from none. Thus it is sufficient to determine the visibility of a particular event from a single representative viewpoint in the cell. We know from the object feature visibility attributes what faces are potentially visible from within a 3-face and what edge-edge occlusions are potentially visible from that same 3-face. To determine which of the potential occlusions actually occur in the projected image we can construct a line from a representative viewpoint in the 3-face to the point of apparent occlusion. If the line intersects any of the potentially visible faces in the interval between the representative viewpoint and the point of apparent occlusion along the first edge then that face prevents the occlusion from being actually visible from any viewpoint in the 3-face. Otherwise, the occlusion is visible from all viewpoints in the 3-face. All of the potentially visible events associated with each 3-face can be tested in this manner. Again, if the actually visible events differ between two 3-faces then they are in different cells, otherwise they are in the same cell.

3.6 Discussion

The algorithm is being implemented on a SUN workstation. The implementation currently produces the correct lattice for any object which does not have e-e-e triplets which generate auxiliary quadric surfaces. Including a variety of display routines which use SunCore graphics, the size of the source program is approximately 11,000 lines of C, representing an executable file of approximately 1 MB in size. The size of the aspect graph data file produced ranges from approximately 10K to 672K for the example objects shown in Table 3. A separate program has been implemented to read the object data files created by PADL-2 and format the boundary surface description of the object for input to the program which creates the aspect graph.

4 Creating the Aspect Graph of Solids of Revolution

Constructing the aspect graph for curved objects is more complex than for polyhedral objects. This is because, in addition to viewpoint-independent lines which arise due to object edges (discontinuities in surface normal), we must also consider viewpoint-dependent lines, called *limbs*. A limb is a contour in the image which arises from a locus of points on the object surface, called a *contour generator*, where the line of sight is tangent to the object surface. Thus the configuration of the line drawing for a particular view of a curved object may be specified by a graph in which the nodes represent junctions in the line drawing and the arcs represent lines due to object edges and limbs. This is basically the *image structure graph* described by Malik [18]. The types of junctions which can occur include those described by Malik [18] and one additional type of pseudo-junction, which we call a transition-S, where the continuous curvature along a line changes between positive and negative. A visual event occurs whenever the image structure graph representing the view of the object changes. Several authors have discussed the classes of visual events which can occur for transparent, piecewise-smooth surfaces under orthographic projection and how they might relate to a partitioning of the viewing sphere [2, 22]. However, the algorithm outlined here is the first for automatically constructing the aspect graph for a defined class of curved objects.

The class of objects which we have chosen for initial study is opaque solids of revolution, where the geometric definition is given as that of a generalized cylinder. This subclass of generalized cylinders has been termed Right, Circular, Straight, Homogeneous, Generalized Cylinders (RCSHGCs) by Shafer [25]. Ponce and Chelberg [21] have recently given an extensive analysis of the properties of RCSHGCs, including the derivation of equations for the contour generators under orthographic and perspective projection. For solids of revolution defined as RCSHGCs, the formation of limbs and the ways in which limbs and edges interact in visual events are rather tightly restricted. It is thus possible to exploit the elements of the RCSHGC definition to formulate an algorithm to generate the orthographic projection aspect graph [5].

To see how the problem might be approached, assume that we have a RCSHGC whose spine is aligned, by convention with the axis of the Gaussian sphere, which represents viewpoint space for the object. First consider the view from a direction orthogonal to the spine (i.e., from some point on the equator of the Gaussian sphere).

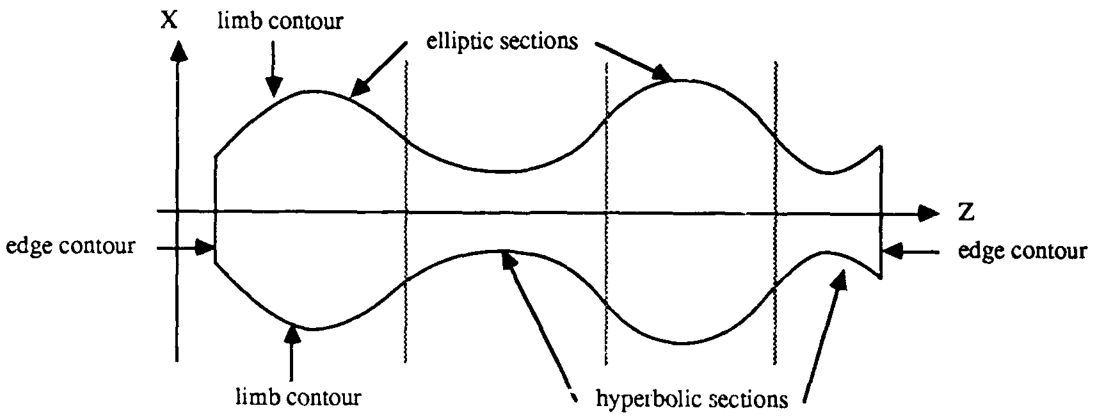


Figure 7 - Side view of solid of revolution depicting types of contours and initial sectioning of object.

The line drawing will be a simple boundary outline, composed of lines for the edges at each of the ends of the object, joined by two limbs generated by continuous contours on the object surface. The shape of the limbs directly corresponds to the sweeping rule of the RCSHGC definition. (See Figure 7.) If the viewing direction is not orthogonal to the spine, then the line drawing becomes more complex, due to the emergence of additional limbs as the contour generators on the object surface become segmented. The problem, then, is to use the RCSHGC definition to enumerate the ways in which the contour generator becomes segmented, as well as the ways in which interactions of the limbs and/or edges occur. Since a RCSHGC is rotationally symmetric about its spine, each visual event will occur for a set of viewing directions which lie along a specific latitude of the Gaussian sphere, and hence can be represented by an angle from the (assumed "north") pole. Thus the partitioning of the Gaussian sphere for the orthographic projection aspect graph of a RCSHGC will take the form of a sequence of latitudinal bands.

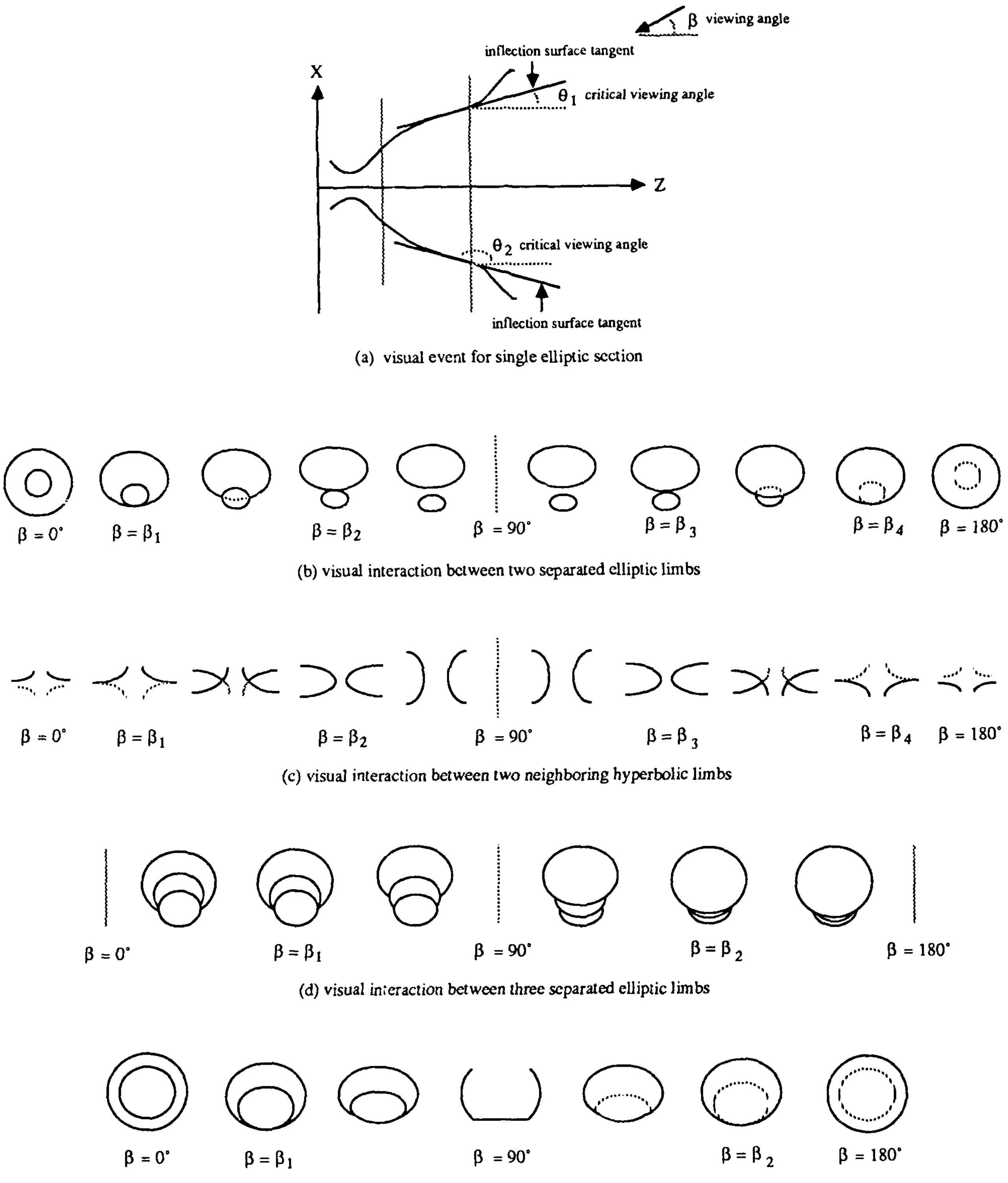
4.1 Enumerating the Primitive Visual Events

To enumerate the set of primitive visual events for RCSHGCs, we must first determine all of the possible individual limbs which can be generated. Then the visual events for each of these limbs in isolation can be calculated. In addition, we consider all possible interactions between pairs of individual limbs, followed by interactions between triplets of limbs, followed by pair and triplet interactions which involve both limbs and edges of the object. The completeness of this set of visual events [illegible] become apparent as they are described.

4.1.1 Individual Limbs

The places on the object surface where individual contour generators segment and join are the transitions between convex (elliptic) and concave (hyperbolic) sections of the object. The contour generator is undefined at points on the object surface where the magnitude of the tangent to the surface is greater than that of the tangent of the viewing angle. Thus the transition points, being places of maximum tangent magnitude, correspond to the first positions that the contour generator becomes undefined as the viewing angle sweeps from the equator to one of the poles. Therefore, the first step in determining the aspect graph for a particular object is to divide it into a sequence of elliptic and hyperbolic sections by analyzing the sweeping rule. The zeroes of the second derivative of the sweeping rule correspond to changes in curvature which divide the elliptic and hyperbolic sections. For a RCSHGC whose sweeping rule is given by a (positive, continuous, single-valued) polynomial of degree N, there are at most N-1 sections defined in this way. (See Figure 7.)

The possible visual events for a limb arising from a single elliptic section of the object, in isolation from limbs arising from other sections, are simply that it may "form" (the viewing angle becomes such that a contour generator exists) or "disintegrate" (the viewing angle becomes such that the contour generator is lost). These events happen only for sections which do not have a local maximum in the sweeping rule within the section. (Sections which do have a local maximum have a contour generator defined over all viewing angles.) The angles at which these events happen are defined by the limits of the range of the surface tangent for the section. For example, for the elliptic section shown in Figure 8.a, viewing angles less than θ_1 and greater than θ_2 have tangent magnitudes less than the smallest surface tangent, and hence no contour generator exists for such viewing angles.



(a) visual event for single elliptic section

(b) visual interaction between two separated elliptic limbs

(c) visual interaction between two neighboring hyperbolic limbs

(d) visual interaction between three separated elliptic limbs

(e) visual interaction between neighboring edge and elliptic limb

Figure 8 - Example visual interactions for single, pairs, and triplets of limbs and edges.

The possible visual events for a limb arising from a single hyperbolic section of the object, in isolation from limbs arising from other sections, are different than for an elliptic section, due to the fact that the limb will "cusp." We will say that a limb (more precisely, the contour generator) *cusps* when it first forms a cusp point (a point at which the tangent of the contour generator is parallel to the viewing direction). Cusping occurs only in hyperbolic sections. For this reason, an analysis similar to the following is not necessary for elliptic sections. The cusping is associated with a particular viewing angle. To one side of this angle, a single continuous limb appears in the image. To the other side of this angle, the visible limb is segmented into two pieces. While the limb has segmented, the contour generator has not. The contour generators of the two visible pieces of limb are actually two sections of the same continuous contour generator, and are connected by an invisible section of the generator. The invisible section is the projection of tangents of the viewing direction to the inside of the object surface rather than the outside. The points at which a visible limb cusps may be found using an equation derived by Ponce [21]. Local maximums of the cusp equation are used to segment the hyperbolic section in a manner analogous to the previous division of the object as a whole. For a RCSHGC with polynomial sweeping rule of degree N there are at most N-1 of these maximums, corresponding to viewing angles at which a visible limb may cusp. However, these are *not* necessarily evenly distributed among the N-1 sections of the object. Successive cuspings in one hyperbolic section will result in multiple visible limb segments for that hyperbolic section, each corresponding to a portion of the contour generator contained within the subsection.

After a hyperbolic section is divided into subsections according to cusping events, the viewing angles of the visual events at which the limbs for each subsection form/disintegrate can be found. The range over which the contour generator of a subsection is defined is determined in the same manner as for an elliptic section. However, the range of potential visibility for the limb is more restrictive in the case of a hyperbolic section. The ends of a limb segment must either be joined with a neighboring limb segment, in which case cusping has not yet occurred, or terminate in a cusp. Thus, if we examine the local minimum of the cusp equation within a subsection, it will determine the lowest viewing angle at which an endpoint of the limb segment exists, which is the angle at which the limb segment forms/disintegrates. During one formation of such a limb, termination junctions (corresponding to the cusp points) will be introduced into the image structure graph.

To summarize, there is potentially a separate contour generator defined for each elliptic and hyperbolic section of the object. For each section, there may be a viewing angle at which the contour generator forms/disintegrates. Additionally, there may be up to N-1 different viewing angles at which cusping first occurs, distributed across all the hyperbolic sections. After subdividing the hyperbolic sections at these cusp points, a possible total of $2 \times N - 2$ different limb segments will have been isolated, which can then interact with each other in the image.

4.1.2 Interactions Involving Pairs of Limbs

A pair of limbs which interact in the image must arise from either (1) contour generators from (sub)sections separated by one or more other (sub)sections, or (2) contour generators from two neighboring (sub)sections of the object.

We *first consider the case of limbs derived from two contour generators from separated sections of the object.* There are three relevant subcases:

1. Limbs from two different elliptic sections.
   Since the object is opaque, if the pair of elliptic sections have a third elliptic section inbetween them, and this third section has a greater local maximum in the sweeping rule than either of the two sections in the pair, then the limbs from the pair cannot interact.

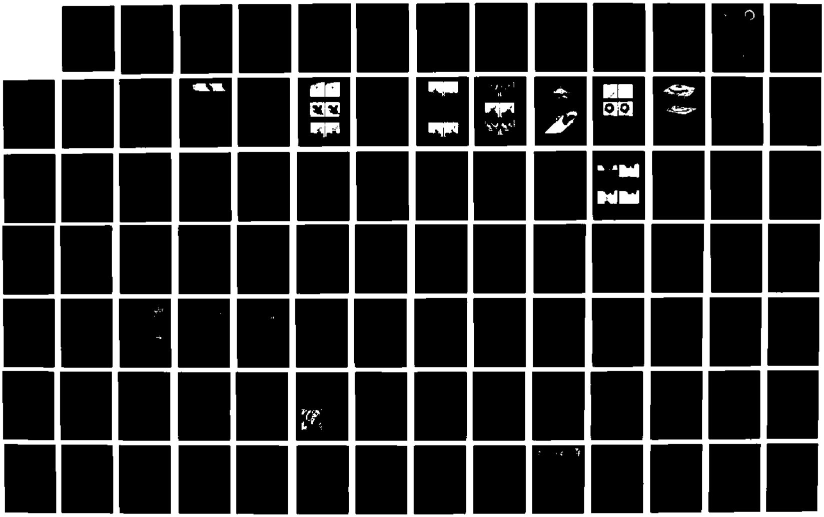
2. One limb from an elliptic section and one from a hyperbolic subsection.
   Similar to the situation with the case of two elliptic sections, if there is an intervening elliptic section with a greater local maximum in the sweeping rule, then the limbs cannot interact.

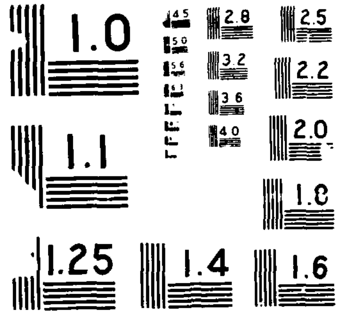
3. Two limbs from subsections of the same hyperbolic section.
   In this case there is no intervening elliptic section to consider. (Limbs from two different hyperbolic sections cannot interact at all, since there must always be an intervening elliptic section which will occlude all of the visual events.)

In all three subcases, the two limbs go from a configuration in which they do not intersect (one may completely occlude the other), through a visual event where they just begin to touch, to a configuration in which they intersect and part of one is occluded. Also, in all three subcases, these interactions result in the formation of T-junctions representing occlusion boundaries. An example of these events for the interaction of limbs from two *elliptic sections is shown in Figure 8.b. The angle pairs β_1, β_4 and β_2, β_3* are supplementary, hence only one from

AD-A212 981 PROCEEDINGS OF IMAGE UNDERSTANDING WORKSHOP HELD IN 10/13
PALO ALTO CALIFORNIA O.. (U) SCIENCE APPLICATIONS
INTERNATIONAL CORP MCLEAN VA L S BAUMAN MAY 89

UNCLASSIFIED N00014-86-C-0700 F/G 17/11 NL




1.0
4.5
5.0
5.6
2.8
2.5
3.2
2.2
3.6
4.0
2.0
1.1
1.8
1.25
1.4
1.6

each pair need be calculated. This is done by setting the coordinates of points along the limbs equal to one another and solving for the intersection condition. Unfortunately, a direct analytic solution does not exist and so an iterative search is necessary.

Now we consider the case of two limbs which arise from neighboring object (sub)sections. Here, there are two relevant subcases:

1. Limbs from neighboring elliptic and hyperbolic sections.

2. Limbs from neighboring subsections within one hyperbolic section.

In the case of limbs from neighboring elliptic and hyperbolic sections, there is a single joining relation. The reason there is no occlusion of one by the other is that the viewing angle at which the joining occurs coincides with the angle at which the limb from the hyperbolic subsection is formed. Prior to the joining, there is a single continuous elliptic limb. On the other side of the visual event, the elliptic limb is split and each half is joined to a limb from the hyperbolic section through the transition pseudo-junction which we call a transition-S. There is also necessarily the introduction of termination junctions corresponding to the cusp points of the hyperbolic limb. The analogous visual event from the supplementary viewing angle represents the disintegration of these junctions. The joining of two limbs from hyperbolic subsections includes an occlusion event prior to the joining event. In this case, the joining does not form a junction, since both limbs have positive curvature. The termination junctions corresponding to the cusps are also disintegrated. The entire process for this type of joining is depicted in Figure 8.c. Occlusion events are determined in the manner mentioned above. The joining of elliptic and hyperbolic limbs occurs at a viewing angle with tangent equal to the surface tangent at the transition point between the elliptic and hyperbolic sections. The joining of hyperbolic limbs occurs at the viewing angle given by the cusp equation at the appropriate maximum.

Now we consider what further visual events might occur between larger groups of limbs. Visual events must be either of the joining variety or the occlusion variety. Visual events of the joining variety cannot occur with groups of limbs greater than pairs, since only two limbs can join together at one point. This leaves the occlusion type of visual event. Each limb in the image has a corresponding contour generator on the object, and all the contour generators occur in some order along the length of the spine. Since the spine of the RCSHGC is, by definition, straight and the cross-sections are, by definition, homogeneous in shape, any one limb in the image may be visibly occluded by at most one other. Therefore the only additional event involving limbs which would change the image structure graph would be when the visible occluder of a limb changes. This event involves a triplet of limbs.

4.1.3 Interactions Involving Triplets of Limbs

A line drawing configuration in which a limb, L_1, visibly occludes part of a second limb, L_2, which in turn visibly occludes part of a third limb, L_3, may go through a visual event where the intersection of L_1 and L_2 projects on top of the intersection of L_2 and L_3, into a new configuration in which L_1 is the visible occluder of both L_2 and L_3. (The projected intersection of L_2 and L_3 is now occluded by L_1.) There are four subcases for which this can happen:

1. Limbs from three elliptic sections of the object.

2. Limbs from two elliptic sections and one hyperbolic (sub)section.

3. Two limbs from within a single hyperbolic section and a limb from an elliptic section.

4. Three limbs from within a hyperbolic section.

Since limbs from two different hyperbolic sections cannot interact, no triplet interaction can exist involving limbs from more than one hyperbolic section. An example of the triplet interaction involving three elliptic limbs is shown in Figure 8.d. In order for the triplet interaction to occur, there must be an intersection of the three ranges in which the individual pair interactions occur. Once again, a search process is required to find the viewing angle at which all three of the limb pair intersections project on top of each other.

4.1.4 Interactions Involving Edges

The edge at each end of the RCSHGC generates an elliptic outline which, being object-dependent rather than viewpoint-dependent, is potentially visible from all viewing directions. Thus there is no visual event in which it forms or disintegrates. Since the outline is elliptic, the edge acts similar to a limb from an elliptic section. For pair interactions, a limb from an elliptic section may be replaced by one of the edges. Thus the additional pair types are edge-elliptic limb, edge-edge, and edge-hyperbolic limb. The edge-edge pair is necessarily an occlusion relation, since the two edges cannot touch on the surface of the object. The other pair interactions involving edges may involve either occlusion or joining events. The joining events give rise to two other junction types, the curvature-L and the three-tangent, depending on the angle from which the join is viewed. The visual events for an edge-elliptic limb interaction of neighboring sections is shown in Figure 8.e. The calculation of visual event angles for occlusion events is done in the same manner as before except that the equation for the edge point coordinates is now used. The tangents of viewing angles for joining events coincide with the surface tangent at the join.

In triplet interactions, one or both edges may again replace elliptic section limbs, with the events then found basically as described before.

4.2 Traversing the Viewing Sphere

As each visual event is enumerated, the angle at which it occurs and the type of the event are recorded, so that we build up a partitioning of the Gaussian sphere into latitudinal bands. Any visual events which are apparently more complex than those enumerated are actually made up of combinations of these primitive events, where the latitudes for the primitive events coincide.

The result of this step is the correct partitioning for a transparent object, given the visual events that we have considered. However, this may be an over-partitioning of the Gaussian sphere for an opaque object, due to some of the enumerated visual events not actually being visible because of occlusion. The relevant partitions for an opaque object may be extracted as follows Begin at one pole of the Gaussian sphere. The line drawing for this view will be a set of concentric circles corresponding to limbs from elliptic sections and the edges. Determine the limbs and edges which generate these visible outlines by checking the sequence of maximums in the sweeping rule. Traverse the sphere toward the other pole, keeping an updated image structure graph for both the transparent and opaque version of the object using the recorded visual events, and eliminating boundary latitudes for events which are not actually visible.

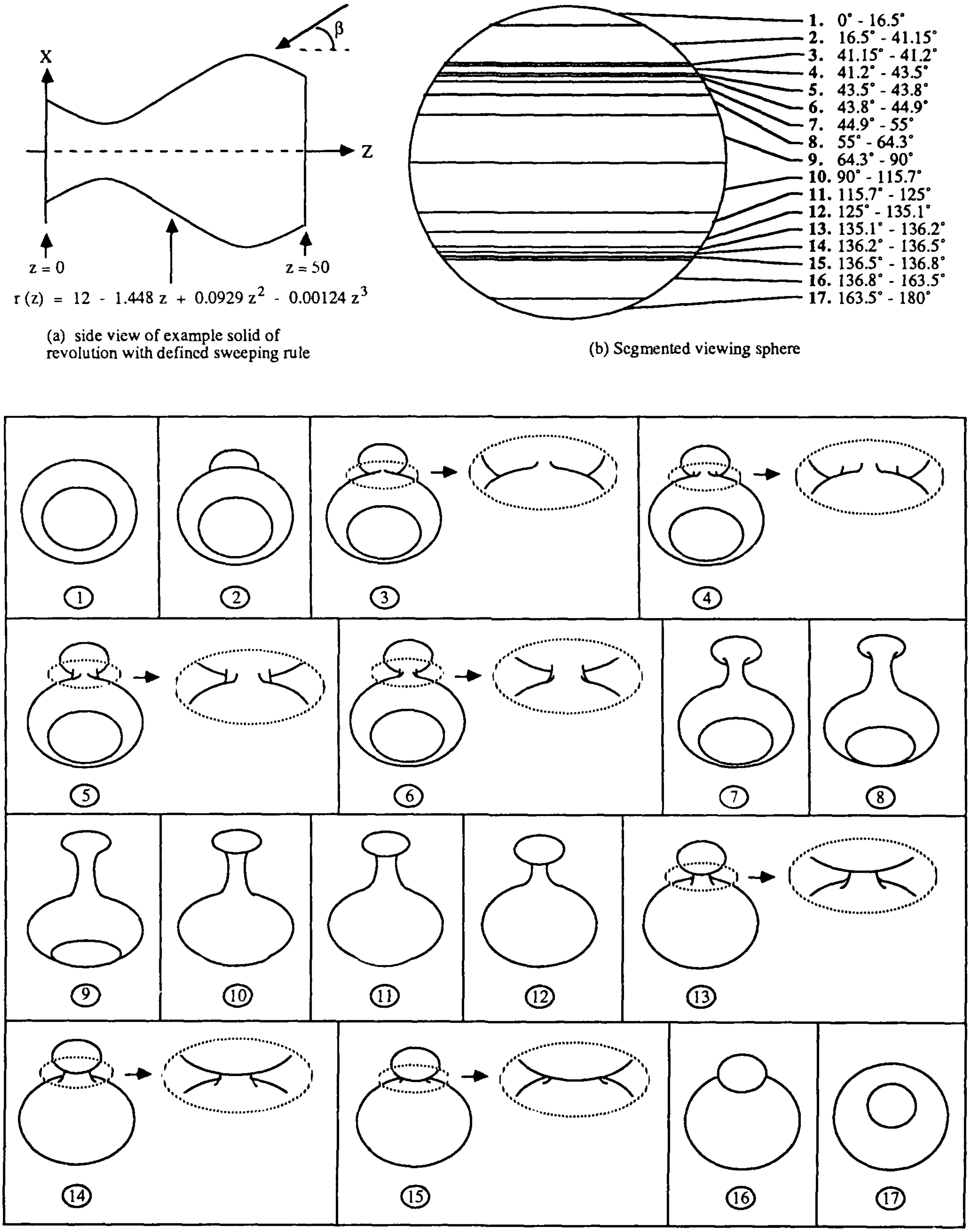
4.3 Discussion

In summary, the algorithm for constructing the aspect graph of a RCSHGC operates as follows. First, divide the object into elliptic and hyperbolic sections. For a RCSHGC with polynomial sweeping rule of degree N, this results in at most N-1 sections. Then further subdivide the hyperbolic sections based on cusping conditions, yielding a maximum total of $2 \times N - 2$ sections. Next, enumerate the visible events for individual limbs. There are clearly at most $O(N)$ of these (exactly, at most $4 \times N - 4$). Next, enumerate all the possible interactions of limb pairs. There are clearly at most $O(N^2)$ of these. Next, enumerate all the possible interactions of limb triplets. There are clearly at most $O(N^3)$ of these. Next, enumerate all the possible limb-edge pairs (at most $O(N)$ of these) and the edge-limb triplets (at most $O(N^2)$ of these). The resulting partitioning of the Gaussian sphere thus has at most $O(N^3)$ latitudinal bands. Finally, traverse the (over-)partitioned viewing sphere to remove the latitudes for occluded visual events and determine the structure of the image structure graph for each aspect. As an example, the results of the algorithm are depicted for a simple flower-vase shape object in Figure 9. As can be seen, this object has a number of different aspects, though many occur only over a very small band on the Gaussian sphere. Examples of most of the event types discussed are demonstrated by this object.

To our knowledge, this is the first algorithm developed to directly calculate the aspect graph for any class of curved objects. Beyond the algorithm itself, we believe that this work is valuable as a demonstration of a general framework for developing such algorithms. Future research topics include generalizing the algorithm to handle a broader class of objects, modeling viewpoint space as 3-D space rather than the Gaussian sphere, and implementing the representation for experimentation with its use in object recognition.

5 Future Research Directions

The main advantages of the aspect graph representation are that (1) it is a well-defined concept in mathematical terms, and (2) it provides a complete summary of the different visual appearances of the object (at least, those which are due to object geometry). We believe that these advantages are quite fundamental, and will be necessary in any object recognition system which is to achieve reasonable performance in a "non-toy" domain of competence.



(a) side view of example solid of revolution with defined sweeping rule

(b) Segmented viewing sphere

(c) Views of object from central viewpoints within each aspect

Figure 9 - Result of aspect graph generation process for flower vase object.

The main disadvantages of the aspect graph representation are that (1) algorithms to create the (exact) aspect graph representation are difficult to develop and implement, and (2) the complexity of the raw aspect graph representation is overwhelming. Since we believe that the advantages are fundamental and necessary, we must look on the (current) disadvantages as the seeds of future research topics. The problem of developing aspect graph algorithms for broader classes of objects is, in fact, difficult. We do not expect the problem to become easy, but we do expect progress from sustained, serious research effort. In the next few years, we expect that it will be possible to automatically produce the aspect graph of objects designed using solid modelers such as PADL-2. The problem of the large complexity of the aspect graph seems daunting and might lead one to consider abandoning the representation altogether. But the large complexity is tied to the visual completeness, and completeness is a fundamental advantage which is not to be dismissed. The true problem is to find ways to efficiently manage the complexity. One possible partial solution is to exploit the fact that not all views represented in an aspect graph are equally likely. Kender and Freudenstein [15] have suggested the idea of probabilities of views based on relative surface areas on the Gaussian sphere. Less formally, Rosenfeld [23] has conjectured that humans achieve "immediate" recognition of "familiar" objects by using a representation with only a small number of the most commonly occurring aspects. Another possible partial solution is to exploit the fact that many views represented individually in the raw aspect graph representation are in fact very similar. Shapiro [26] discusses a possible method of defining *view classes* which can have different levels of similarity. Many of these ideas converge in the suggestion of some sort of hierarchical aspect graph representation. This representation would represent the subdivision of viewpoint space by the "biggest" or "most important" visual events at its highest level, and successively finer subdivisions at lower levels. (Again, this echoes some of Rosenfeld's conjecture [23].) The hierarchical representation would aid in handling the time complexity of searching a large database. If the representation could be dynamically expanded, then the problem of handling the space complexity is also aided. Perhaps the most important question here is just what principle(s) should be used to establish the hierarchy.

References

[1] Burns, J.B. and Kitchen, L.J. 1987. Recognition in 2D Images of 3D Objects from Large Model Bases Using Prediction Hierarchies, IJCAI, 763-766. See also 1988 DARPA IUW, 711-719.

[2] Callahan, J. and Weiss, R. 1985. A Model for Describing Surface Shape, CVPR, 240-245.

[3] Castore, G. 1984. Solid Modeling, Aspect Graphs, and Robot Vision, in Solid Modeling by Computer, Pickett and Boyse, ed., Plenum Press, New York, 277-292.

[4] Edelsbrunner, H., O'Rourke, J. and Seidel, R. 1986. Constructing Arrangements of Lines and Hyperplanes with Applications, SIAM Journal of Computing 15, 341-363. See also 1983 Symp. on Found. of CS, 83-91.

[5] Eggert, D. and Bowyer, K.W. 1989. Computing the Orthographic Projection Aspect Graph of Solids of Revolution, USF Computer Science and Engineering Technical Report.

[6] Gigus, Z, Canny, J. and Seidel, R. 1988. Efficiently Computing the Aspect Graphs of Polyhedral Objects, ICCV '88, 30-39.

[7] Gigus, Z. and Malik, J. 1988. Computing the Aspect Graph for Line Drawings of Polyhedral Objects, CVPR, 654-661. See also 1988 Robotics and Automation proceedings, 1560-1566.

[8] Goad, C. 1984. Special Purpose Automatic Programming for 3-D Model-Based Vision, DARPA IUW, 94-104. See also Fast 3-D Model-Based Vision, in From Pixels to Predicates, Pentland, ed., Ablex Publishing (1986), 371-391.

[9] Hansen, C., and Henderson, T. 1988. Towards the Automatic Generation of Recognition Strategies, ICCV '88, 275-279.

[10] Hansen, C., and Henderson, T. 1987. CAGD-Based Computer Vision, IEEE Workshop on Computer Vision, 100-105.

[11] Hebert, M. and Kanade, T. 1985. The 3D-Profile Method for Object Recognition, CVPR, 458-463.

[12] Hemami, H., Weimer, F.C., and Advani, J.G. 1975. Identification of Three Dimensional Objects by Sequential Image Matching, IEEE Conference on Computer Graphics, Pattern Recognition, and Data Structures, 273-278.

[13] Ikeuchi, K. 1987. Generating an Interpretation Tree from a CAD Model for 3-D Object Recognition in Bin-Picking Tasks, International Journal of Computer Vision, 145-165. See also Ikeuchi, K and Kanade, T. Automatic Generation of Object Recognition Programs, IEEE Proceedings 76, 8, 1016-1035 (August 1988).

[14] Korn, M.R. and Dyer, C.R. 1987. 3-D Multiview Object Representations for Model-Based Object Recognition, Pattern Recognition 20 (1), 91-103.

[15] Kender, J.R. and Freudenstein, D.G. 1987. What Is A "Degenerate" View?, IJCAI, 801-804. See also 1987 DARPA IUW, 589-598.

[16] Koenderink, J.J and van Doorn, A.J. 1979. The Internal Representation of Solid Shape with Respect to Vision, Biological Cybernetics 32, 211-216.

[17] Lowe, D.G. 1980. Solving for the Parameters of Object Models from Image Descriptions, DARPA IUW, (April 1980), 121-127.

[18] Malik, J. 1987 Interpreting Line Drawings of Curved Objects, IJCV 1, 1, 73-103.

[19] Plantinga, H. and Dyer, C. 1986. An Algorithm for Constructing the Aspect Graph, Foundations of Computer Science, 123-131.

[20] Plantinga, H. and Dyer, C. 1987. Visibility, Occlusion and the Aspect Graph, CS Tech. Report #736, University of Wisconsin.

[21] Ponce, J. and Chelberg, D. 1987. Finding the Cusps and Limbs of Generalized Cylinders, Int. J. of Computer Vision 1, 195-210. See also 1987 Robotics and Automation proceedings, 62-67.

[22] Rieger, J. 1987. On the Classification of Views of Piecewise Smooth Objects, Image and Vision Computing, 91-97.

[23] Rosenfeld, A. 1987. Recognizing Unexpected Objects: A Proposed Approach, 1987 DARPA IUW, 620-627.

[24] Swain, M. 1988. Object Recognition from a Large Database Using a Decision Tree, 1988 DARPA IUW, 690-696.

[25] Shafer, S.A. 1985. Shadows and Silhouettes in Computer Vision, Kluwer Academic Publishers, Boston, Massachusetts.

[26] Shapiro, L.G. 1988. Viewing Clusters for Object Localization, SPIE #938: Digital and Optical Shape Representation, 408-418.

[27] Stark, L., Eggert, D., and Bowyer, K.W. 1988. Aspect Graphs and Non-linear Optimization in 3-D Object Recognition, ICCV '88, 501-507. See also USF Technical Report CSE-88-0016.

[28] Stewman, J.H. and Bowyer, K.W. 1987a. Implementing Viewing Spheres: Creating the Aspect Graph of Convex Polyhedral Objects, SPIE #786: Applications of Artificial Intelligence, 526-532.

[29] Stewman, J.H. and Bowyer, K.W. 1987b. Aspect Graphs for Planar-Face Convex Objects, IEEE Workshop on Computer Vision, 123-130.

[30] Stewman, J.H., Stark, L., and Bowyer, K.W. 1988. Restructuring Aspect Graphs Into Aspect- and Cell-Equivalence Classes for Use In Computer Vision, in Lecture Notes in Computer Science #314: Graph Theoretic Concepts in Computer Science, Springer-Verlag, 230-241.

[31] Stewman, J., and Bowyer, K.W. 1988. Creating the Perspective Projection Aspect Graph of Polyhedral Objects, ICCV '88, 494-500.

[32] Watson, L.T. and Shapiro, L.G. 1982. Identification of Space Curves from Two-Dimensional Perspective Views, IEEE Transactions on Pattern Analysis and Machine Intelligence 4, 5 (September 1982), 469-475.

[33] Watts N. 1988. Calculating the Principal Views of a Polyhedron, ICPR, 316-322. See also CS Tech. Rep. 234, University of Rochester, December 1987.

[34] Werman, M., et al. 1986. The Visual Potential: One Convex Polygon, CS-TR-1690, University of Maryland. To appear in CVGIP.

Qualitative shape from stereo

Daphna Weinshall
CBIP, MIT, E25-201, Cambridge MA 02139

Abstract

Vision is sometimes described as a problem of inverse optics, which makes its solution mathematically cumbersome and unstable. Human vision does not seem to involve the computation of the inverse mapping of the projection of 3D world onto a 2D retina but something more qualitative. This work concentrates on qualitative shape information that can be obtained from stereo disparities with little computation. Local surface patches are classified as convex, concave, hyperbolic or parabolic, using a simple function of image disparities. The axes of minimum, maximum, and zero curvatures are also obtained. The algorithm works well with synthetic images and exact disparities. It is used to compute axes of zero curvature on a real image.

1 Introduction

When two different cameras record the same scene, objects are projected into different locations in each image, depending on the relative locations of the two cameras with respect to each other and to the object. This disparity in position between the two images may be used to obtain the exact coordinates of the object if the relative transformation between the two cameras coordinate systems is known. This view of stereo vision regards the problem as a problem of inverse optics, namely, the goal is to find the inverse transformation of the optical process (perspective projection). Thus stereo is divided to two main subproblems. The first is the correspondence, matching the two images to find the appropriate disparity in position for each object or feature. The second is the determination of the distance to objects in space using geometrical transformation.

The matching problem turned out to be very difficult to solve for the general case. Thus most of the research concerning stereo vision deals with new algorithms or approaches to obtain and improve the matching (e.g., [1], [2] and [3]). The transformation between stereo disparity and depth requires the knowledge of the current viewing geometry. This part is often ignored, assuming the imaging geometry is known or can be computed a priori using test images for calibration. The calibration problem, the computation of the imaging geometry from images disparities, is generally difficult (possibly more so than the matching problem) and involves the solution of a nonlinear set of equations with many unknowns (see [4]). Constraining the geometry allows for simpler solutions to this problem (e.g., [5]). It would not be surprising if biological vision avoid solving the calibration problem in light of its computational difficulty. Human depth perception is rather approximate, so if the inverse optics (that is, exact depth) is being calculated, this information is not retrievable.

Recently some doubt has been cast on the need for inverse optics (see, e.g., [6], [7] and [8]). More qualitative approaches to navigation have been proposed (see [9] and [10]). An alternative theoretical approach to the analysis of stereo and motion (assuming matching is given) has been proposed by Koenderink and Van-Doorn (e.g., [11], [12] and [13]). They show how various qualitative properties of objects and the motion field can be obtained from invariants of vector fields (the optical flow or disparity field). Such results shed light on characteristic quantities that could be easily derived from disparities and optical flow. These results, however, are derived using vector field analysis and therefore the existence of a differentiable vector field is assumed (though singularities are addressed by the authors in different ways). The meaning of such results for a discrete image and vector fields is not always clear.

The purpose of this work is to explore possible qualitative shape information that can be obtained directly and simply from disparities without further computation (namely, independent of the imaging geometry). The classification of local surface patches as convex, concave, parabolic (cylindrical), hyperbolic (saddle point) or planar is computed from image disparities only (in fact only disparities in polar angles are needed). The directions of the axes of principal curvature are obtained as well as the axes of zero curvature when they exist. The analysis does not depend upon special constraints on the nature of objects in the environment, such as assuming smoothly curved surfaces or a particular analytic representation of the surface. The results hold for a family of surfaces that correspond to some continuous interpolation between the points considered for the calculations. The classification may not give the "correct" differential type of a surface patch when the surface changes irregularly between the points where disparities are given. An earlier version of the algorithm is used to compute axes of zero curvature for cylindrical objects in a real image of cans at various orientations.

2 Local shape descriptors

The curvature of any curve on a regular surface through some point can be written as a linear combination of two principal curvatures κ_1 and κ_2. These are the curvatures of two perpendicular curves on the surface that obtain the extrema of the curvatures of all curves on the surface passing through the same point. If κ_n is the normal curvature of some curve on the surface and θ is its angle with the first principal axis, then

$$\kappa_n = \kappa_1 \cdot \cos^2\theta + \kappa_2 \cdot \sin^2\theta \ . \tag{1}$$

Thus the local behavior (curvature) of a local surface patch can be described in terms of two numbers, κ_1 and κ_2. The product of these two numbers $\kappa_1 \cdot \kappa_2$ is called the Gaussian curvature and its sign characterizes the local structure of the surface in an intuitive way. Distinct surface types are defined as follows:

1. elliptic ($\kappa_1 \cdot \kappa_2 > 0$)
   - convex, see figure 1a-left ($\kappa_1, \kappa_2 > 0$)
   - concave, see figure 1a-right ($\kappa_1, \kappa_2 < 0$)
2. parabolic (cylindrical), see figure 1b ($\kappa_1 \cdot \kappa_2 = 0$, $\kappa_1 > 0$ or $\kappa_2 < 0$)
3. hyperbolic (saddle point), see figure 1c ($\kappa_1 \cdot \kappa_2 < 0$, namely $\kappa_1 > 0$ and $\kappa_2 < 0$)
4. planar ($\kappa_1 \cdot \kappa_2 = 0$, $\kappa_1 = \kappa_2 = 0$)

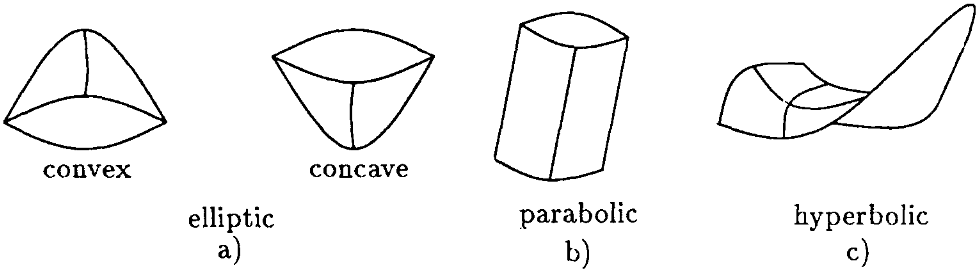


Figure 1: An illustration of the different surface types used for classification of surfaces, see text.

It follows from equation (1) that the number of asymptotes, or the number of curves on the surface with zero-curvature, determines the local behavior of the surface. Specifically:

1. elliptic: no asymptote
2. parabolic: one asymptote
3. hyperbolic: two asymptotes
4. planar: infinite number of asymptotes

Thus for surfaces where the asymptotes are locally straight lines on the surface, the number of straight lines on the surface that cross a point will determine the local behavior of the surface. As will be shown in section 4, the decision whether a straight line in the image originated from a straight line on the surface can be made given image coordinates only, without the need to compute exact 3D coordinates, and with very simple calculations.

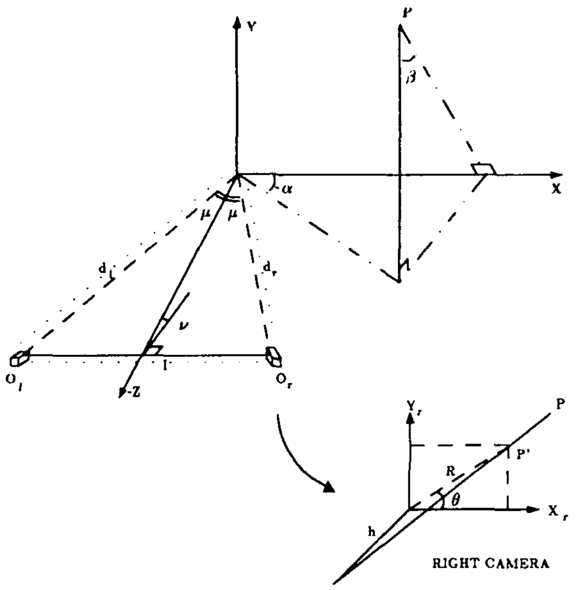


Figure 2: Above, the 3D coordinate system defined by two cameras. Below, the image plane of the right camera.

3 Basic geometrical derivations

Given two cameras, assume that the principal rays intersect at a fixation point. Also, assume that the epipolar plane of the fixation point (the plane through the principal rays of the cameras, henceforth "base plane") includes the X-axes of both cameras. Thus rotation about the principal rays of the cameras is fixed. We will use the following coordinate system (see figure 2): let the fixation point be the origin, the base plane (which passes through this point) be the $X - Z$ plane, and the line perpendicular to this plane through the origin be the Y-axis. On the $X - Z$ plane, the principal rays of both cameras intersect at the origin and create an angle 2μ between them. Let the Z-axis be the angle-bisector of 2μ, and the X-axis perpendicular to the Z-axis. A similar system can be defined for motion if the fixation point is kept constant, that is, the cameras follow a single object.

For a given point P let α denote the angle of tilt and β denote the angle of slant (see figure 2). Thus the Cartesian representation of P is $(\frac{z}{\tan\alpha}, \frac{z}{\tan\beta}, z)$, where z is its depth relative to the fixation point in the above coordinate system. Let (x_l, y_l) and (x_r, y_r) be the Cartesian coordinates of the projection of P on the left and right images respectively (see lower part of figure 2). Using polar coordinates, the two projections can be written as (R_l, ϑ_l) and (R_r, ϑ_r) respectively. Then the following holds (see [14]):

$$\tan\alpha = \frac{1}{\tan\mu} \cdot \frac{\cot\vartheta_r - \cot\vartheta_l}{\cot\vartheta_r + \cot\vartheta_l} \ , \quad \tan\beta = \frac{\cot\vartheta_r - \cot\vartheta_l}{2\sin\mu}$$

Thus, the two angles α and β (of P) depend only on the angle of convergence 2μ and the polar angle of the projection (P') on each image. It can be shown that the polar angles are preserved under projection, through any point on the principal ray, onto either a spherical body (like the eye) or a planar one (a camera). There is no dependence on other parameters of the cameras, their relative positions or the angle of gaze ν.

4 Computation of local shape descriptors from disparities

Let P_0 and P_i be two points on the surface of some object. $\vec{P_0} = z_0(\frac{1}{\tan\alpha_0}, \frac{1}{\tan\beta_0}, 1)$ and $\vec{P_i} = z_i(\frac{1}{\tan\alpha_i}, \frac{1}{\tan\beta_i}, 1)$. Let

$$\vec{N^i} = \vec{P_0} \times \vec{P_i} \qquad (2)$$

N^i is perpendicular to the plane passing through P_0, P_i and the origin (the fixation point), assuming they are not collinear. This plane is not related to the surface we wish to characterize. However, If two points P_i and P_j

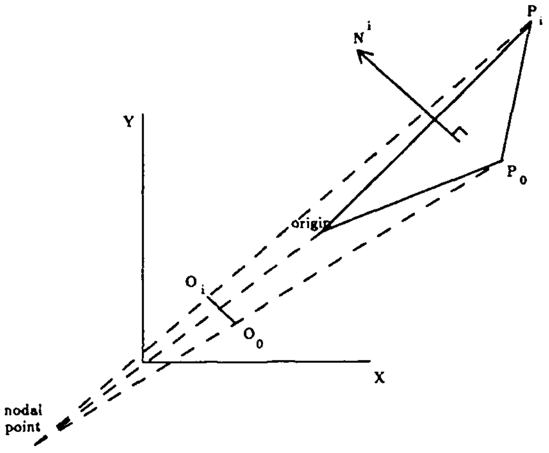


Figure 3: Two points on the surface P_0 and P_i are projected to points O_0 and O_i on the image. $\vec{N}^i = \vec{P}_0 \times \vec{P}_i$ is perpendicular to the plane passing through P_0, P_i and the origin (the fixation point).

are collinear with P_0, by definition $\vec{N}^i = \vec{N}^j$. Thus N^i can be used to determined whether three points that are collinear in the image originate from a straight line (an asymptote) or not. Since we know that a straight line is always projected to a straight line in the image, $\vec{N}^i = \vec{N}^j$ is a necessary and sufficient condition for being an axes of zero curvature.

Therefore we have shown the following: Let O_0, O_1 and O_2 be the projections of P_0, P_1 and P_2, where O_1 and O_2 lie on different sides of O_0. Then P_1, P_0 and P_2 are on an asymptote (are collinear in space) if and only if two conditions hold:

1. O_0, O_1 and O_2 are collinear in the image,
2. $N_x^1 = -N_x^2$ and $N_y^1 = -N_y^2$.

The second condition can be verified using the polar angles only of the projections of P_0, P_1 and P_2 onto the two images. This follows from the following expression for N^i:

$$\vec{N}^i = z(\frac{1}{\tan\mu} N_x^i(\vartheta_l^0, \vartheta_r^0, \vartheta_l^i, \vartheta_r^i) , \frac{1}{\sin\mu} N_y^i(\vartheta_l^0, \vartheta_r^0, \vartheta_l^i, \vartheta_r^i) , 1)$$

where

$$N_x^i = \frac{(\cot\vartheta_l^i - \cot\vartheta_l^0) - (\cot\vartheta_r^i - \cot\vartheta_r^0)}{(\cot\vartheta_r^i - \cot\vartheta_r^0) + (\cot\vartheta_l^i - \cot\vartheta_l^0)} , \quad N_y^i = \frac{\cot\vartheta_r^i \cot\vartheta_l^0 - \cot\vartheta_l^i \cot\vartheta_r^0}{(\cot\vartheta_r^i - \cot\vartheta_r^0) + (\cot\vartheta_l^i - \cot\vartheta_l^0)} \tag{3}$$

The computation of asymptotes can be made more robust in that it can be verified from other cues beside stereo. Pentland ([15]), for example, has argued that $d^2 I = 0$ along an axes of zero curvature, where I is the intensity. This information may also be deduced from motion (straight lines in motion remain straight) or linear perspective. In any case, using condition (2) or either of these methods to count the number of zero-curvature axes is extremely sensitive to noise since they require counting zero crossings.

Let P_0 be a point on the surface whose neighborhood we wish to characterize. Let P_1 and P_2 be two points whose projections are collinear with the projection of P_0 in the image and which lie on different sides of P_0 as we had before. Let

$$\Delta N_x = N_x^1 + N_x^2.$$

ΔN_x is the difference in the x component of N^1 and N^2 and, once again, it depends only on the polar angles of the projections of P_0, P_1 and P_2 on the two images (see equation (3)). (The plus in the definition is due to the fact that N^1 and N^2 are defined with opposite signs by the vector multiplication in equation (2) since O_1 and O_2 lie on different sides of O_0.) It can be shown that the sign of ΔN_x is equal to the sign of the normal curvature of a smooth curve passing through P_0, P_1 and P_2. We approximate this normal by the angle bisector of the angle created at P_0 by connecting it to P_1 and P_2. More specifically, the following proposition can be proved (proof is omitted):

proposition: Let θ_0 be the slope of O_0 in the image (its polar coordinate). Let ΔN_x^θ be ΔN_x defined for P_0, P_1 and P_2 such that the slope of the line in the image passing through their projections O_0, O_1 and O_2 is θ. Then for all directions θ around O_0, $\theta_0 < \theta < \theta_0 + 180°$:

$$\begin{array}{ll} \Delta N_x^\theta < 0 \Rightarrow & \text{negative normal curvature} \\ \Delta N_x^\theta = 0 \Rightarrow & \text{zero normal curvature} \\ \Delta N_x^\theta > 0 \Rightarrow & \text{positive normal curvature} \end{array}$$

The signs reverse (negative ΔN_x^θ implies positive normal curvature and vice-versa) if $\theta_0 - 180^o < \theta < \theta_0$. This proposition holds as long as the points P_0, P_1 and P_2 are not collinear with the origin ($\theta \neq \theta_0$).

Thus it is possible to classify the surface around point P_0 using the following algorithm. For each direction θ around O_0, $\theta_0 < \theta < \theta_0 + 180^o$, choose two points in the image O_1 and O_2 on both sides of O_0 that are collinear with each other and define a slope θ. It is assumed that O_1 and O_2 are the projections of points lying on the same surface as P_0. Compute ΔN_x^θ for each θ, thus defining a function of θ to be denoted by $D(\theta)$. Then:

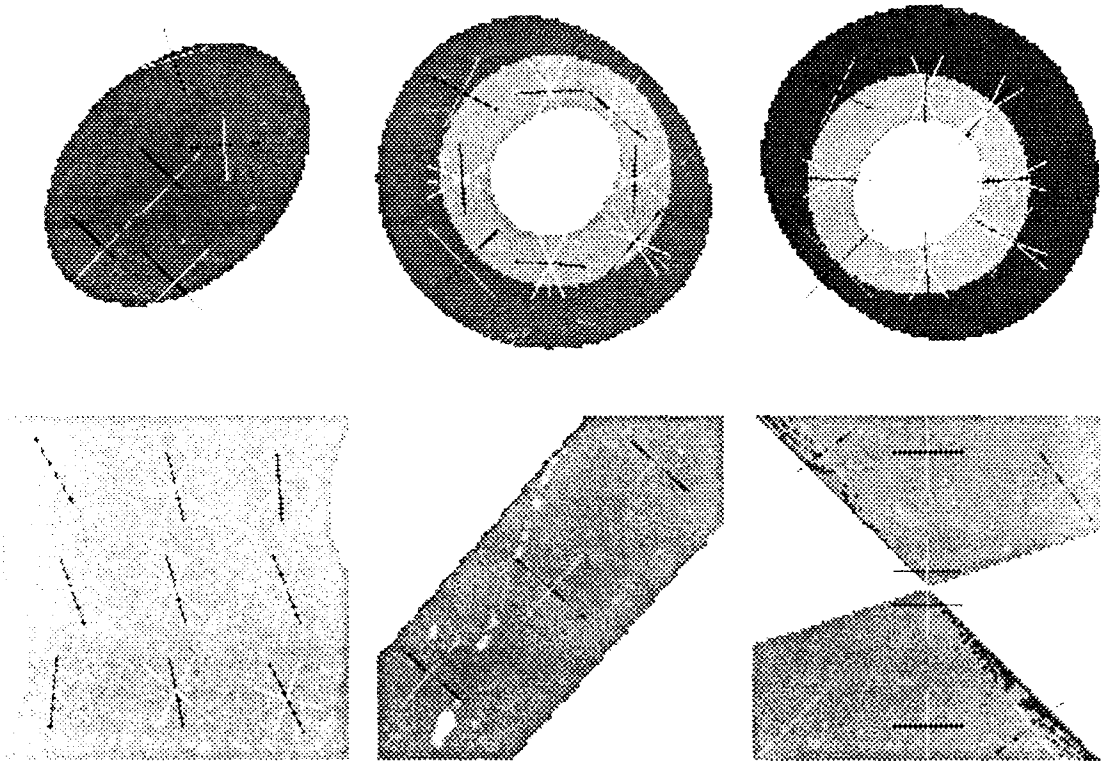
- $D(\theta) = 0 \ \ \forall\theta \ \ \Rightarrow$ surface is planar.
- $D(\theta) > 0 \ \ \forall\theta \ \ \Rightarrow$ surface is convex.
- $D(\theta) < 0 \ \ \forall\theta \ \ \Rightarrow$ surface is concave.
- $D(\theta) \geq 0 \ \ \forall\theta$ or $D(\theta) \leq 0 \ \ \forall\theta \ \ \Rightarrow$ surface is parabolic (cylindrical). The axis of zero curvature is the axis for which $D(\theta) = 0$.
- $D(\theta)$ changes sign $\Rightarrow$ surface is hyperbolic. In this case the asymptotes are the directions for which $D(\theta) = 0$. The principal directions (direction of minimum and maximum curvature) are the lines that cross the two angles defined by the asymptotes.

In the presense of noise some threshold should be used instead of 0, which may cause regions whose curvature is low to be all classified as planar. Then the directions of the axes of zero curvature and principal axes cannot be computed exactly.

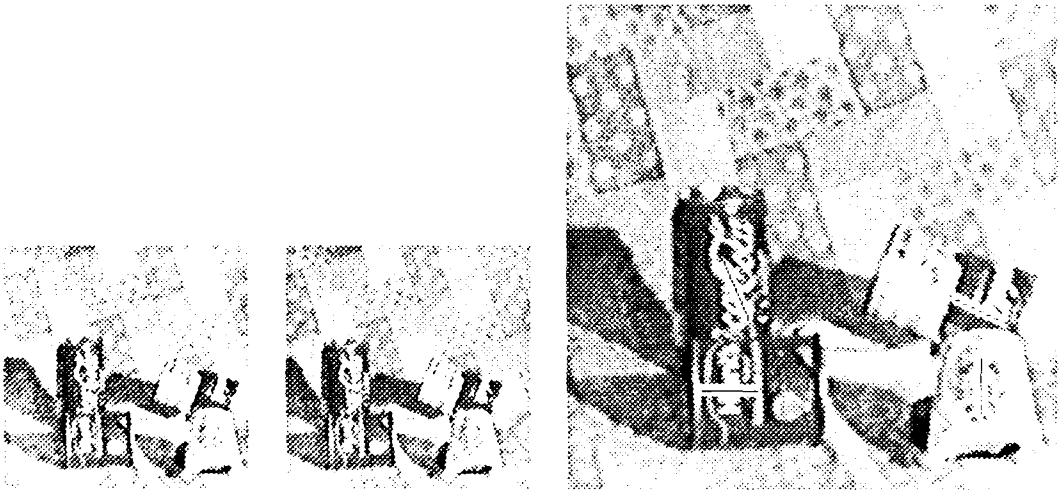
5 Examples

Figure 4 shows the results as obtained for synthetic data of a torus, a cylinder, a cone, a hyperbola and a sphere. The shadings are explained in the legend of the figure. The results are accurate both for surface classification and directions of important axes. Note the emergence of the parabolic line on the torus (the line separating the hyperbolic region from the convex/concave region, whose type is parabolic). It is often argued that these parabolic lines are important for image representation (see [16]).

An early version of this algorithm (see [14]) has been used to compute axes of zero curvature for known cylindrical objects in a real image of cans at different orientations (see figure 5). In this example the camera was moved manually to obtain a stereo pair. The fixation point (and hence the origin of each image coordinate system) was taken to be the center of the right image anu the corresponding point in the left image (which in the above "bad" example is a few pixels to the left of the center of the left image). The two 256x256 images have been matched using a parallel motion algorithm implemented on the Connection Machine ([17]). Its output has been smoothed by averaging with a 3x3 window over neighboring pixels. In a fixed region at the center of each object, the direction of the zero-curvature axis has been estimated at each pixel. The direction obtained by the largest number of pixels in the region was selected as a final estimate. In an image containing four cylindrical objects at various orientations, the true axis of zero-curvature has been obtained for three (figure 5). A rather good approximation has been obtained for the fourth, where the "second best" direction has been selected (we have used a rather coarse quantization of directions). Additional errors may occur if the central region is not chosen "appropriately", that is, if it lies too close to the boundary of an object or if it covers area with little texture.



[illegible] torus; right: inside of torus. Second row, left: a hyperb[illegible] [illegible] The shadings mean the following: [illegible] lightest grey marks hyperbolic regions (internal rings and [illegible] the hyp[illegible]) [illegible] darker grey marks convex regions [illegible] external ring of inner torus). [illegible] lines of minimum curvature or maximal negative [illegible]



[illegible] white lines [illegible]

6 Summary

This work has been motivated by two observations. First, in order to obtain precisely a surface $z = f(x, y)$ it is necessary to solve first for the imaging geometry (calibration) for each image pair. The calibration problem is generally difficult, possibly more so than the matching problem usually considered the core of the stereo problem. Second, humans apparently do not compute the exact depth from stereo disparities and probably do not compute exactly the imaging geometry. Therefore we have looked for some more qualitative information that can be obtained from disparities only and with few additional computations. One measure of the behavior of local surface patches is obtained directly from disparities with a simple computation. It is the classification of surfaces as convex, concave. cylindrical, planar or hyperbolic. In addition, directions of principal curvature axes and asymptotes are obtained.

References

[1] David Marr and Tomaso Poggio. A computational theory of human stereo vision. *Proceedings of the Royal Society of London B*, 204:301–328, 1979.

[2] H.H. Baker and T.O. Binford. Depth from edge and intensity based stereo. In *Proceedings IJCAI*, pages 631–636, Vancouver, 1981.

[3] W. Hoff and Narendra Ahuja. Extracting surfaces from stereo images: An integrated approach. In *Proceedings of the International Conference on Computer Vision*, pages 284–294, June 1987.

[4] B. K. P. Horn. *Robot Vision*. MIT Press, Cambridge, Mass., 1986.

[5] H.C. Longuet-Higgins. A computer algorithm for reconstructing a scene from two projections. *Nature*, 293:133–135, 1981.

[6] W. B. Thompson and J. K. Kearny. Inexact vision. In *Workshop on Motion, Representation and Analysis*, pages 15–22, May 1986.

[7] Alessandro Verri and Tomaso Poggio. Against quantitative optical flow. In *Proceedings of the International Conference on Computer Vision*, pages 171–180, June 1987.

[8] S. Edelman and T. Poggio. Representations in high-level vision: reassessing the inverse optics paradigm. In *Proceedings Image Understanding Workshop*, April 1989.

[9] R. C. Nelson and J. Aloimonos. Using flow field divergence for obstacle avoidance: towards qualitative vision. In *Proceedings of the International Conference on Computer Vision*, pages 188–196, December 1988.

[10] Hanspeter A. Mallot, Heinrich H. Bulthoff, and James J. Little. Interaction of different modules in depth perception. In *Arvo annual meeting abstract issue*, page 398, May 1988.

[11] J. J. Koenderink and A. J. van Doorn. Invariant properties of the motion parallax field due to the movement of rigid bodies relative to an observer. *Optica Acta*, 22(9):773–791, 1975.

[12] J. J. Koenderink and A. J. van Doorn. Local structure of movement parallax of the plane. *J. Opt. Soc. Am.*, 66:717–723, 1976.

[13] J. J. Koenderink and A. J. van Doorn. Geometry of binocular vision and a model for stereopsis. *Biol. Cybernetics*, 21:29–35, 1976.

[14] D. Weinshall. Qualitative depth from stereo, with applications. Technical Report AI-Memo 1007a, Artificial Intelligence Laboratory, Massachusetts Institute of Technology, 1987.

[15] A P. Pentland. Local shading analysis. In A. P. Pentland, editor, *From pixels to predicates*, pages 40–77. ablex, New Jersey, 1986.

[16] A. L. Yuille. Zero crossings on lines of curvature. *Computer Vision, Graphics, and Image Processing*, 45:68–87, 1989.

[17] James J. Little and Heinrich Bulthoff. Parallel computation of optical flow. Technical Report AIM-929, Artificial Intelligence Laboratory, Massachusetts Institute of Technology, 1987.

ACCURATE SURFACE DESCRIPTION FROM BINOCULAR STEREO*

Steven D. Cochran and Gérard Medioni
Institute for Robotics and Intelligent Systems
School of Engineering
University of Southern California
Los Angeles, California 90089-0273

ABSTRACT

We present a Stereo Vision System which attempts to achieve robustness with respect to scene characteristics, from textured outdoor scenes to environments composed of highly regular man-made objects. It offers the advantages of both area-based (dense map) and feature-based (accurate disparity) processing by combining them whenever possible. In the current version, the area-based process occurs first and is refined by the integration of edge information. It is based on our observation that whenever there is enough "texture" (measured as intensity variation in a small window), then a correct correspondence can be obtained by a local process. The area-based process therefore proceeds by computing a texture measure for each image view and performing a simple cross correlation between them. A match is accepted if both views agree on a peak and this peak is high enough. The resulting dense disparity image with a few holes and incorrect matches is then filtered using the smoothness assumption to fill small gaps and remove small spikes. Note that contrary to the case of feature-based stereo, this smoothness assumption is justified since we reason about patches of opaque objects, and that we can make inferences about occlusion and detect "penumbral" areas (visible in only one of the views). This disparity map is a smoothed version of the true one, however, because of the finite width of the windows used in processing. The problem is most acute at C_0 (depth) and C_1 (crease) discontinuities, but can be solved by introducing the edge information: the disparity map is adaptively smoothed subject to the constraint that the disparity at edgels is fixed. It is important to note that this method gives an active role to edgels parallel to the epipolar lines, whereas they are discarded in most feature-based systems. We have obtained very good results on complex scenes in different domains.

1 INTRODUCTION

Humans are able to perceive depth in 2-D "monocular" images and in an enhanced form by the stereoscopic combination of a pair of images. The process used by the human visual system to do this, however, is not well understood. Much research has been devoted to the automatic abstraction of information about objects in images, in order to produce autonomous systems which are able to "perceive," and operate upon their environments and, in some cases, to gain insight about the operation of the human visual system. In 1982 Barnard and Fischler [1] defined six steps necessary to stereo analysis: Image acquisition, camera modeling, feature acquisition, image matching, depth reconstruction, and interpolation.

Of these steps, image matching is widely considered to be the most difficult to solve, and is clearly dependent on the choice of feature primitives. Given two views of a scene, a correspondence must be established between those points which are visible in both scenes. When the matched features are low-level and dense, such as the intensity, we call the matching strategy an "area-based" process; while for sparse and usually more abstract features, such as edge-segments, we use the term "feature-based." Some systems, like ours, use a hybrid approach with multiple features both sparse and dense.

The problem of matching the selected features is made difficult by several problems, the most important of which are:

- Photometric variation at a point viewed from two different angles,
- Occlusion of some points in the image, *visible only from one view,*
- Presence of a repetitive texture pattern, and
- Lack of texture in a region.

*This research was supported by the Defense Advanced Research Projects Agency under contract F33615-87-1436, monitored by the Air Force Wright Aeronautical Laboratories.

In the system we are building we attempt to provide answers to each one of these problems:

To overcome the photometric variation, as well as some of the digitization noise, we need a feature which is as invariant as possible to changes in viewpoint from two different angles. However, since we desire a dense match, we also need a dense feature. We have chosen a measure of local intensity variation, since this will be less subject to photometric variation and digitization noise while also providing a simple measure of local texture.

To overcome the problem presented by occlusion, we utilize the fact that those points that are not hidden or out-of-scene must, necessarily, be visible from both views. Thus if the two views do not agree on the location of the point then either one of them is wrong, or the point is visible only from one camera (and therefore most likely also wrong, since there should exist no possible stereo correspondence). The few points for which the two views agree on an incorrect disparity are discarded through the use of the smoothness assumption, and by imposing an order constraint on the final match.

In general, errors due to repetitive texture are more difficult to handle, as they require a global rather than local analysis. We currently treat such false matches, as above, by the application of an order constraint. Other approaches, such as using the edge matches to provide this disparity limit (and a "more global" match estimate), or multi-level processing to give a local "vergence," should be used in order to provide a better solution.

In the case of insufficient texture, where there is no possible match, it is important to know where such areas are and to mark them as being unmatchable locally (as opposed to forcing a random match).

In the remainder of this paper, we start by giving a brief summary of related systems, then present our approach in detail and illustrate the step-by-step processing on a stereo pair of the Renault part: First we apply a local variation operator to the individual intensity images, then perform a cross-correlation on the result from which the "best" peaks are selected. The smoothness assumption and an order constraint are used to detect and correct errors in this initial disparity map, subject to the verification by agreement from each of the two views. While this gives a good area-based match, we use the (monocular) edgels from the two views to further correct for "blurring" across the discontinuities. Finally, we are able to accurately label visible and hidden points, and to apply a final interpolation. We then present reconstructions of the original scene.

In the end, we present our conclusions as to the usefulness of the above ideas to stereo analysis and our plans for further integration of area and feature based stereo strategies.

2 RELATED WORK

Area-based methods have been applied successfully to the analysis of aerial terrain images, where the surface varies smoothly and continuously. Such systems, however, often have difficulty with scenes that contain orientation and depth discontinuities. This is because the correlation windows that cross surface discontinuities cannot usually be correctly matched. In addition, the area-based techniques are more sensitive to noise than the feature-based ones. Most area-based methods make the assumptions that the intensity-texture nearby a point in the scene is invariant from both points of view and that each such neighborhood is detectably different from other neighborhoods in the scene. These methods break down when either of the above two assumptions are false within the search region. Surface discontinuities and photometric variations can cause the first assumption to be false, while the second is violated when the scene has either a lack of texture or when the texture repeats through the scene. The area-based methods offer the advantage of directly generating a dense disparity map.

Mori, *et. al.* [2] attempted to make the correlation adaptive by varying the size of the window and they used edgels to verify the prediction. Hanna [3, 4, 5] showed improvement in the cross-correlation by using dense features abstracted from the intensity data, and has implemented a complete system for stereo processing with little or no operator intervention.

The currently preferred approach in the vision research community is to match more abstract features, rather than match texture regions in the two images, since such features are less sensitive to noise. Feature-based analysis provides more precise positioning (for the feature) in the individual images and it can attain correspondingly higher accuracy for its correspondences in 3-D (Arnold [6, 7]). The most commonly used features are points along the edges of intensity discontinuities. These points, termed edgels for edge-elements, are useful because they represent the points at which most of the scene information is available. However, the feature-based methods provide only sparse matches and require interpolation as well as some method for modeling occlusion. In addition, the feature-based process may be confused by large local change in disparity, and it is very difficult to incorporate the smoothness assumption into the matching strategy since it is most likely to be violated at edges.

Marr and Poggio [8] proposed a computational model of human stereo vision, using zero-crossings in the Laplacian of the Gaussian of the intensity, as a matching feature. They suggest that three constraints should be satisfied in choosing global correspondence: Compatibility, Uniqueness, and Continuity. The latter is the same as the Figural Continuity constraint proposed by Mayhew and Frisby [9]. Grimson [10] implemented an improved version of this model, which gives good results when there is a sufficiently dense set of features, but cannot deal very well with real images or when discontinuities occur. Arnold [6], and Baker [11] used various forms of the Viterbi dynamic programming algorithm to match edges, and Ohta and Kanade [12] extended Baker's inter-scanline search, again using dynamic programming, to find an optional matching surface (however their three-dimensional search is very expensive). Medioni and Nevatia [13] match linear edge segments which cut through the epipolar lines and thus automatically insure inter-scanline continuity. Their matching is based on a minimum differential disparity criterion which attempts to preserve the local disparity and thus enforce a smoothness assumption. Hoff and Ahuja [14] attempt to combine the feature matching, contour detection, and surface interpolation into one process. Their results are very impressive, but they fail when their matching feature (zero-crossings) are too sparse, also they cannot accurately locate the contours.

In both area and feature based stereo correspondence methods, it is often necessary to interpolate values for those regions for which no disparity can be found. Some methods such as those by Hoff and Ahuja [14] and Boult and Chen [15] have combined the interpolation into the normal processing. In addition to interpolating, these methods are also used to smooth surfaces and to isolate discontinuities. Grimson [10] interpolated surfaces from sparse depth data by fitting a surface which represents a minimization of what he called "quadratic variation," Terzopoulos [16] extended this approach by compiling a complete computational theory of visible-surface representations. He attempted to locate discontinuities by locating significant inflection points on the resultant surface. Blake and Zisserman [17] introduced their "Graduated Non-Convexity" algorithms for the weak membrane and the weak plate. These allow the search for depth discontinuities and orientation discontinuities respectively. Saint-Marc and Medioni [18] present another method which smooths the surface while preserving discontinuities and which facilitates the detection of discontinuities.

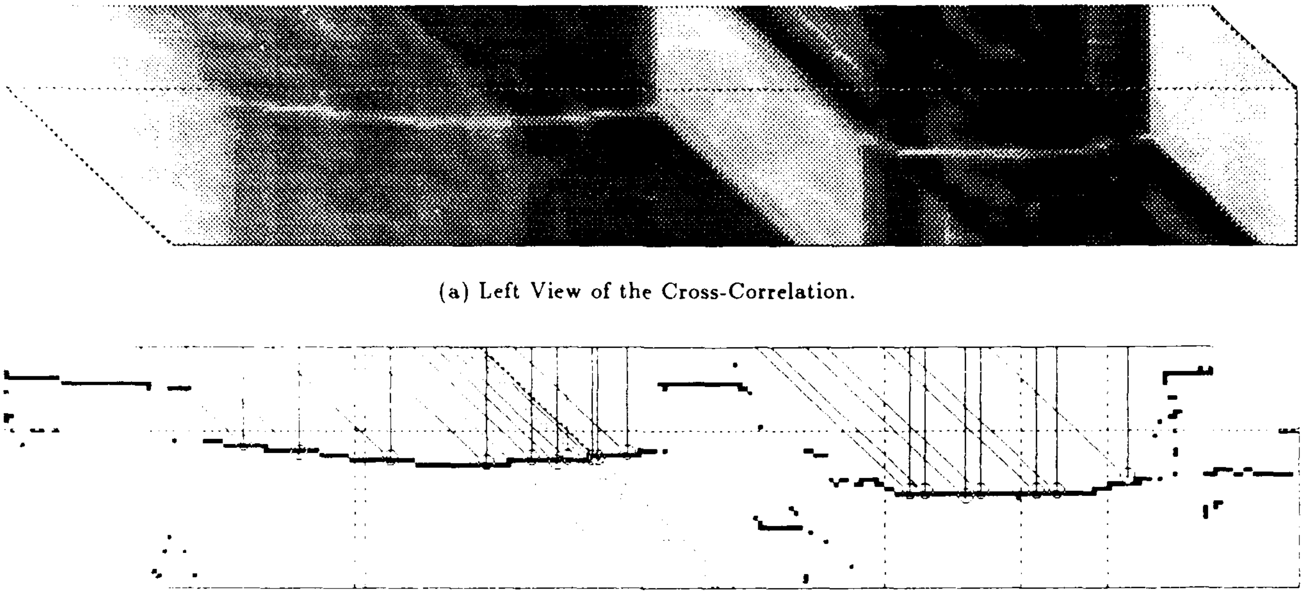
Any of these interpolation methods may be used to extract the discontinuities in the manner that we suggest, by adding a strong preference for the existence of the discontinuity contour (either depth or orientation) to occur at edgels or along edges.

3 DESCRIPTION OF THE METHOD

The methodology used in this research can be understood from the analysis of the pictures in Figure 1. This figure shows the cross-correlation of a pair of corresponding rows (row 191, see Figure 2) from the Renault Part image. The peaks of the cross-correlation (bright areas) represent the best matches. Figure 1b shows the extracted peaks which have been overlaid with the matched (solid) and unmatched (dashed) edgels (see Figure 6). These two images show the line-of-sight from the two images so that the left image view is vertical, while the right image view crosses it at an angle of -45°. The horizontal dashed line through the center corresponds to zero disparity (that is, the point at which the crossing lines of view are from the same column of each image). By looking at Figures 1a and b it should be clear to the reader that, for most points, the correct disparity can be found by simply extracting the peaks in the cross-correlation array. Onc this is done, we need only to focus our attention on the problem areas.

The first occurs in areas without measurable texture. For these, it is important simply not to generate matches from the cross-correlation information. Where multiple matches (peaks) occur, we prefer the highest peak that both views can agree on. It is possible (although unlikely) for the correct peak not to be selected, so we also must impose a smoothness constraint on the surfaces and we assume that order reversals cannot occur due to the spatial shift between viewpoints. These last two constraints also correct most of the "wrong" matches. The remaining unmatched areas represent points that are either occluded in one of the views or "visible" points for which the two views cannot agree. Finally, there is the "blurring" beyond the actual edge, which can be seen in Figure 1b.

The entire processing, which we illustrate on the Renault part of Figure 2, is therefore as follows: First we compute our matchable "feature," and generate the cross-correlation for the entire image. Then we extract peaks, subject to agreement from both views, this provides us with the correct matching at most textured points that are visible from both views. Next we apply a smoothness assumption and the ordering constraint and remove the conflicting "matches." Next we attempt to fill the gaps in the image from the lesser peaks in the cross-correlation and repeat our checks until we cannot improve the matching. Finally we extract the edgels from the original intensity image and use them as (monocular) cues as to the location of possible depth discontinuities during an adaptive smoothing process which trims surfaces which have overrun into the wrong surface. This gives us a very good set of matches, which



(a) Left View of the Cross-Correlation.

(b) Correlation Peaks and Edge Matches.

Figure 1: Cross-Correlation slice (Row 191 of the Renault Pair in Figure 3). The corresponding edge matches from the same row in Figure 6 are shown in (b).

we can further improve by labeling all of the points as being visible or not, and marking the depth discontinuity contours. The following subsections present a detailed account of these steps.

3.1 AREA-BASED PROCESS

The area-based processing matches the texture in the image pair to produce a dense disparity map. By performing the matching both from left-to-right, then again from right-to-left and finally requiring that the two solutions agree, we can obtain a much better estimate.

Choice of Primitive

We developed the idea of using a local variation operator because intensity was not sufficiently invariant to photometric changes and to digitization. Local variance has been used in the past, but did not give a very robust measure. We use Equation (1) which provides a real value between 0, where there the intensity is constant, to a theoretical maximum of 1 where there is the most variation (the actual value is a little less). Also, since it is possible for the intensity to be zero, the special case of all zero values in a window is defined to be zero. Empirical analysis has shown that a 5 × 3 window is a good size, providing good sampling without causing too much smoothing.

$$V[I] = 1 - \frac{\bar{x}}{\sqrt{(\frac{n-1}{n})\sigma^2 + \bar{x}^2}} \qquad (1)$$

where

$\bar{x}$ = mean of the intensity values in a local window

σ^2 = standard deviation of the intensity values in the window

n = number of pixels in the window (we use 5 × 3)

Equation (1), like simple variance, still does not provide clear peaks when the local variation is small (such as the background of Figure 2). So it does not yet reflect the ideal feature that we desiie. One alternative is to adaptively adjust the cross-correlation, but instead, we choose to flatten the histogram of the above measure of variation. We prefer to use a local (16 × 16) window for a parallel machine, and the entire image on a serial machine. Both methods give good results and produce the desired feature for matching using the cross-correlation. One other adjustment was found to be necessary for robustness, and that was that we should know when there is too little texture for the cross-correlation to give a reasonable match. This is difficult to do with the equalized values, so we use a threshold, usually 10^{-6}, to mask off the locally unmatchable values.

Figure 2 shows the Renault part stereo pair.[1] The image is 256 x 256 x 8 and the disparity ranges from -30 (near) to 15 (far) pixels. The right image has been adjusted to bring the image points into alignment with the corresponding image points in the left image on the same scan lines.

Figure 3 shows the Renault part stereo pair after being filtered using the local variation operator. Note the enhancement of the very slight texture in the background and on the table while preserving sufficient texture to match the Renault part. These images are easier to free-fuse than the original, but are slightly blurred.

Correlation and Peak Extraction

We generate the cross-correlation using a normalized cross-correlation bounded by manually supplied minimum and maximum overall disparity. In the special case where there are no features within the correlation window, a value of zero is assigned.

Figure 1a shows a slice through this cross-correlation volume at row 191 from the bottom which cuts through the two lobes of the Renault part. Figure 1b shows the peaks selected from this slice provided that the value at that point:

1. Is less than or equal to predecessor and successor values along the view line-of-sight.
2. has a magnitude of at least half that of the largest peak along this path.

Two-Views

The advantage of using two views along with the enforcement of opacity and uniqueness assumptions has been discussed before by Drumheller and Poggio [19], and Yuille and Poggio [20]. We extend this notion beyond the enforcement of a "forbidden zone," and use it to first clean up the image by removing conflicts (points not in agreement from both views and those generating order-reversals), and then fill-in unmatched points using the correlation peaks.

First, the peaks, extracted from the cross-correlation, are used to construct an initial match estimate as shown in Figure 4 by selecting a single maximum peak from each view such that both views agree as to its disparity. This initial estimate is a good, but noisy, estimate of the disparity. Most of the noise occurs in those areas which are visible in only one of the views.

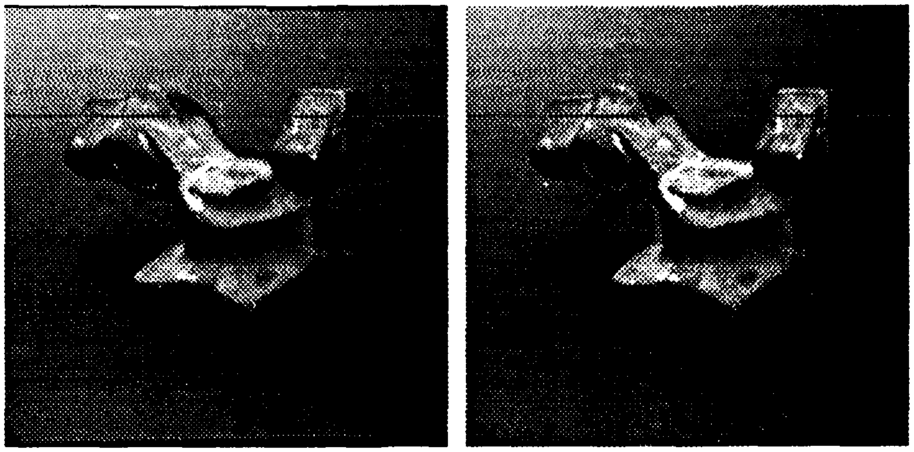
Next we remove those pixels which violate the smoothness assumption or the order constraint. The gaps are then filled with cross-correlation peaks which are exactly in agreement by both views and which adjoin existing agreed upon points (these may not be the maximum peaks). We repeat this process, but this time we fill with peaks which are within a bounded (e.g. ±1 pixel) agreement from both views and which adjoin existing points. A final repeat step completes the process but now fills with non-peak points which are within a bounded (e.g. ±1 pixel) agreement from both views and which adjoin existing points. Figure 5 shows the results of this process starting from the initial matches in Figure 4. This figure represents an impressive result for an area-based process, but does not have the accuracy that we desire due to the "blurring" across the discontinuity edges.

3.2 DISPARITY-MAP REFINEMENT

The area-based disparity maps define a dense surface which is reasonably accurate, except for a tendency to "blur" beyond the object contours (the more textured surface leaks into the less textured region). This can be refined using edge information. We can use edgels (or edges) from any source. For this paper we have used edgels from the Canny Edge Detector [21]. Figure 6 shows the edges resulting from the intensity images in Figure 2.

Instead of using edgels matched by an independent feature-based process (which could give rise to possible conflicts), we use edgels in the 2-D image and associate to them the disparity obtained from the area-based process. Since edges are likely to correspond to depth or surface orientation discontinuities, we smooth the disparity map, keeping the disparity at the edgels fixed. This is implemented using the adaptive smoothing formalism developed Saint-Marc and Medioni [18]. All points whose disparity shifts by more than a constant amount (we use 1.0 pixels) are discarded, removing the "blurred" fringe around the actual contour edges. Note that we are using the edgels as monocular

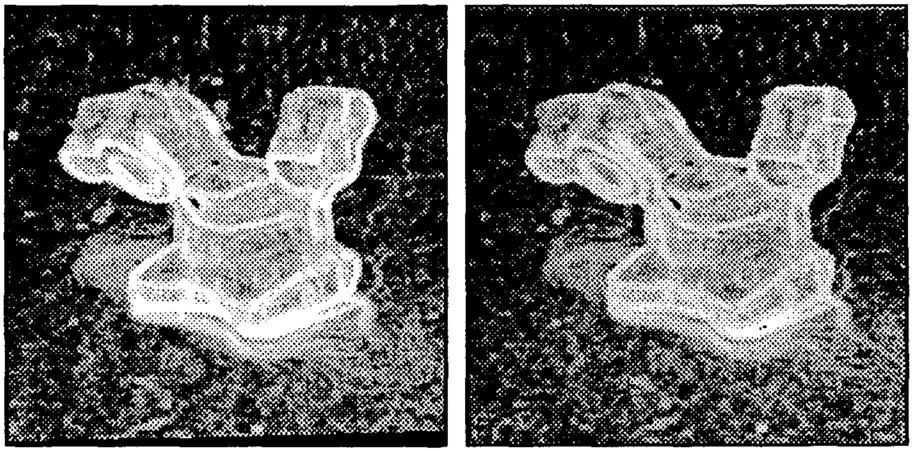
[1] Throughout this paper, the left and right images of stereo pairs have been reversed for free fusing.



(a) Intensity (Right)

(b) Intensity (Left)

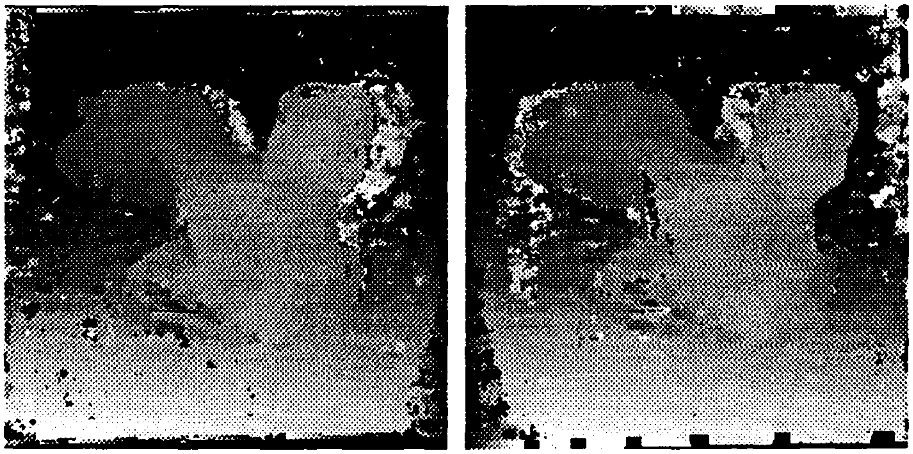
Figure 2: Renault intensity image pair, with row 191 highlighted.



(a) Variation (Right)

(b) Variation (Left)

Figure 3: Local variation from Figure 2



(a) Disparity (Right)

(b) Disparity (Left)

Figure 4: Initial disparity estimate derived by selecting the best peaks of the cross-correlation of the images in Figure 3.

cues, so that even the edges with orientation close to the epipolar line directions play an active role, whereas they are discarded in all feature-based approaches.

Figure 7 illustrates the result of this process on the results of the refined disparity map in Figure 5. From this, we can now label the pixels into four groups, as listed below and shown in Figure 8, and mark the depth discontinuities.

1. Known disparity values are white.
2. Penumbral points, visible only from this view, are black.
3. Points which were not, but should be matchable are light gray.
4. Points visible only from this view because of image clipping in the other view (or initial masking) are dark gray.

Given this data we now attempt to reproduce the original object as shown in Figure 2. First we interpolate along those regions labeled in Figure 8 as being visible. This interpolated disparity image is shown in Figures 9ab. The shaded representations of these are shown in Figures 9cd. These were produced by assigning the surface a simple reflectance function and positioning an imaginary light in space.

Figure 9e is a 3-D plot of the left-image-disparity data from Figure 9b plotted at each fourth pixel. The disparity ranges from -16 (far) to +31 (near) and the zero-plane of the plot is moved backward 27 from the zero-plane of the image. A rendered view is also shown in Figure 9f in which the original intensity values (Figure 2b) were projected onto the surface rotated 45° about the Y-axis and scaled ×7 in the original Z-axis.

4 ANOTHER RESULT

Here we demonstrate the results of the Stereo Vision System on an an aerial view of the Pentagon, shown in Figures 10ab. This stereo pair was was obtained from Professor Takeo Kanade of the Carnegie-Mellon University Computer Science Department. The image is 512 x 512 x 8 and the disparity ranges from -9 (near) to 8 (far) pixels. Figures 10c-f show the results following the integration of the area and feature based processing. The results are quite impressive, but also point out two weaknesses with the method. The first is that when the disparity difference between two surfaces is only about 1 pixel, they cannot be accurately separated as can be seen around the underpass of the bridge in the upper left corner. Also, when two edges are very close together, the "blurring" can extend past more than one edge. We have no solution for this except to work at a higher level of resolution. Note that the occluded areas around the walls are well localized except for a false match in the area-based processing of the upper right corner. Figure 10g shows a 3-D plot of the Figure 10d and Figure 10h shows a rendered view of the same figure.

5 CONCLUSION

We have presented some key ideas whose use in standard stereo correspondence provides some impressive results on several images. In particular, we show one use of those edgels which are aligned parallel to the epipolar lines which, along with all edgels, serve as loci to limit the search for discontinuities in depth or orientation. We have also introduced a new low-level feature, which allows standard cross-correlation to automatically adapt to local levels of variation. Finally, we have demonstrated an important paradigm in stereo matching: the agreement by two views for each disparity value to resolve the ambiguity and verify the matches.

These have allowed us to produce a system which infers more than the disparity at each point, we can also accurately label those points as to whether or not they are visible from one or both views and to mark those boundary contours which form the depth-discontinuity. This methodology is robust. When there are no edge features, it acts as a good area-based process, and when there is no texture, as a feature based system.

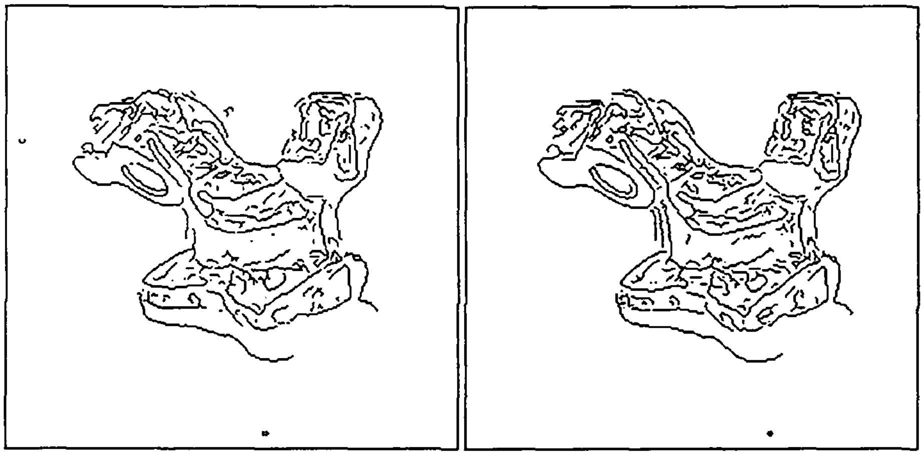
6 FUTURE RESEARCH

So far we have only used the simplest integration of area and feature-based methods. We plan to extend the interaction using abstract monocular and stereo cues to guide the area-based stereo matching. We especially want to reduce the possibility of false matches and blurring. Also, we feel that the orientation discontinuities should, now, be nearly as easily detectable as the depth discontinuities.

Another area that we feel needs more work is the determination of the best approach for interpolation through those areas of the image which cannot be matched due to having too little texture.

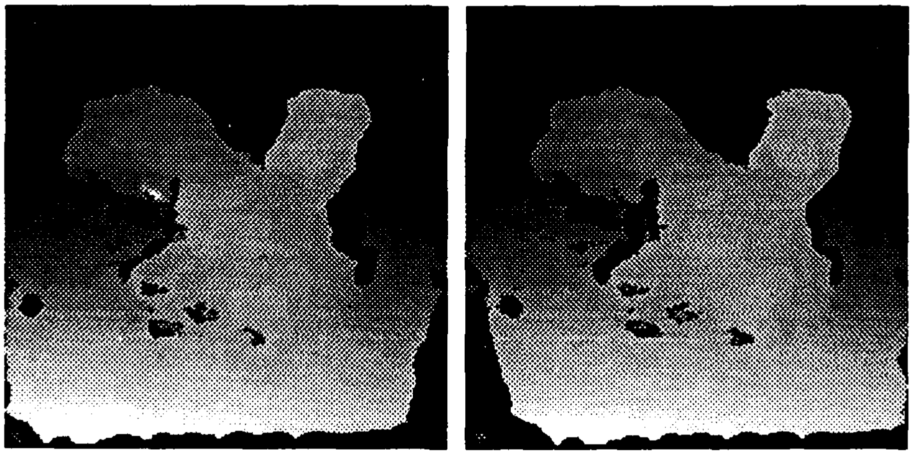
(a) Smoothed (Right) (b) Smoothed (Left)

Figure 5: Refined disparity map after smoothness and order constraints.



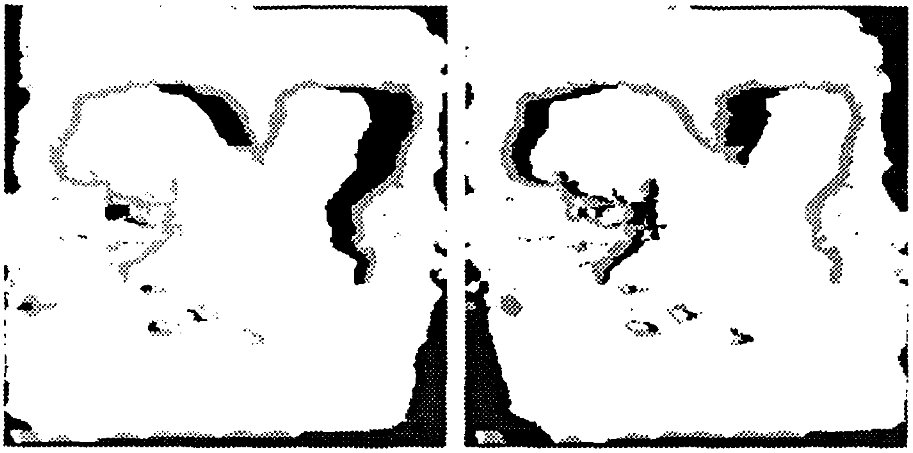
(a) Edges (Right) (b) Edges (Left)

Figure 6: Edges from Canny's operator applied to the images of Figure 2.



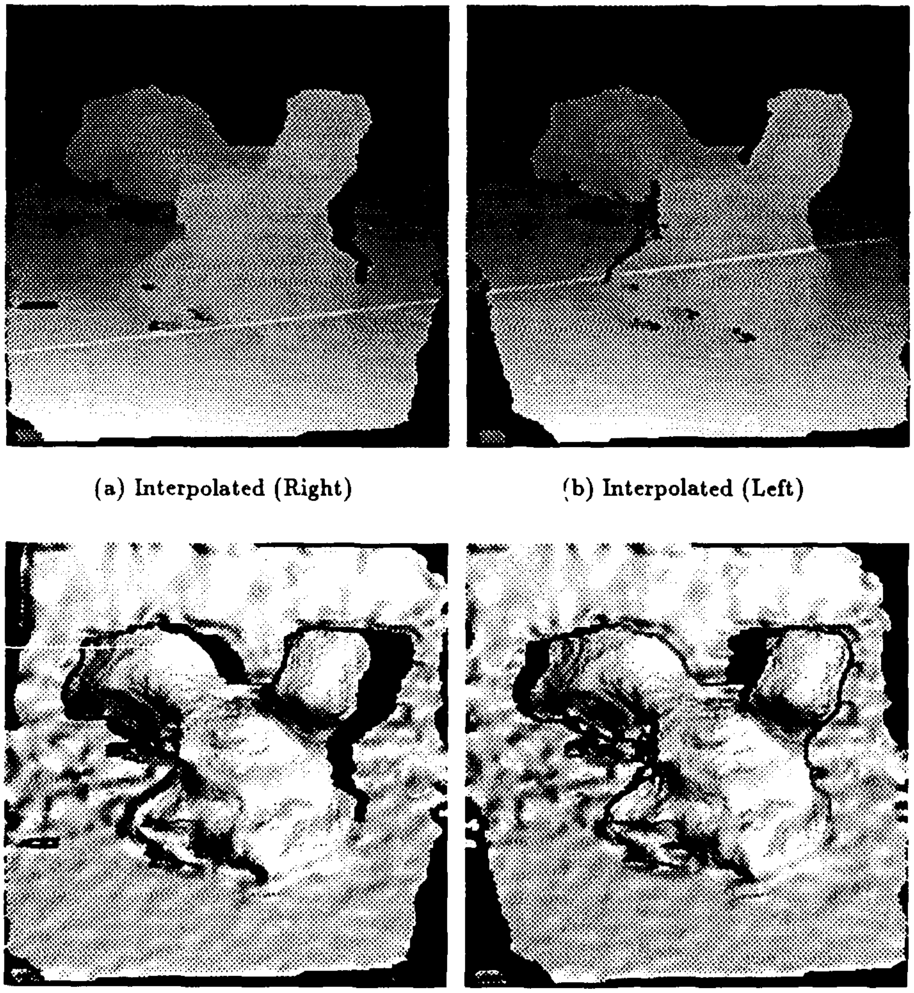
(a) Integrated Results (Right) (b) Integrated Results (Left)

Figure 7: Disparity map after incorporation of edge information.



(a) Labeled Regions (Right) (b) Labeled Regions (Left)

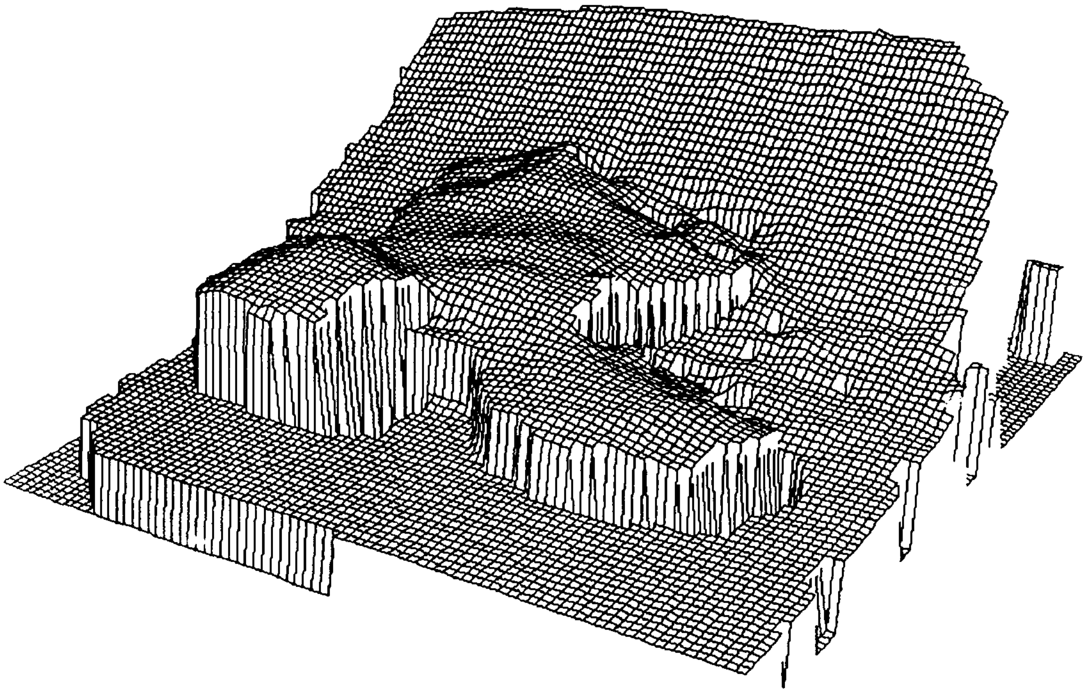
Figure 8: Labeled points from Figure 7.



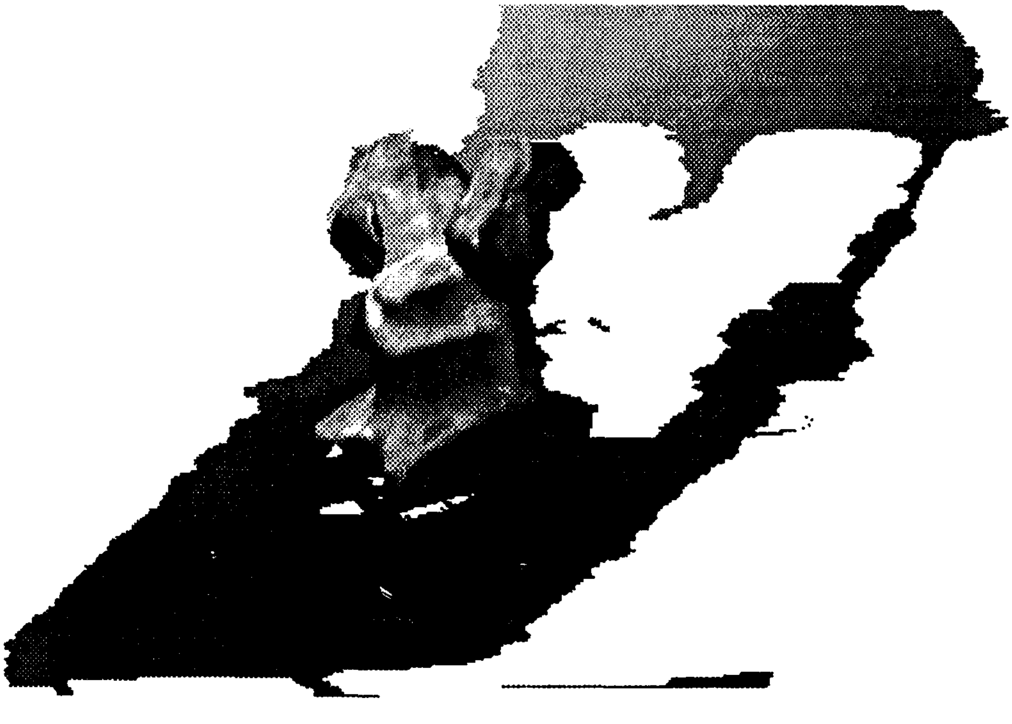
(a) Interpolated (Right) (b) Interpolated (Left)

(c) Shaded (Right) (d) Shaded (Left)

Figure 9: Final disparity maps and their shaded representation.

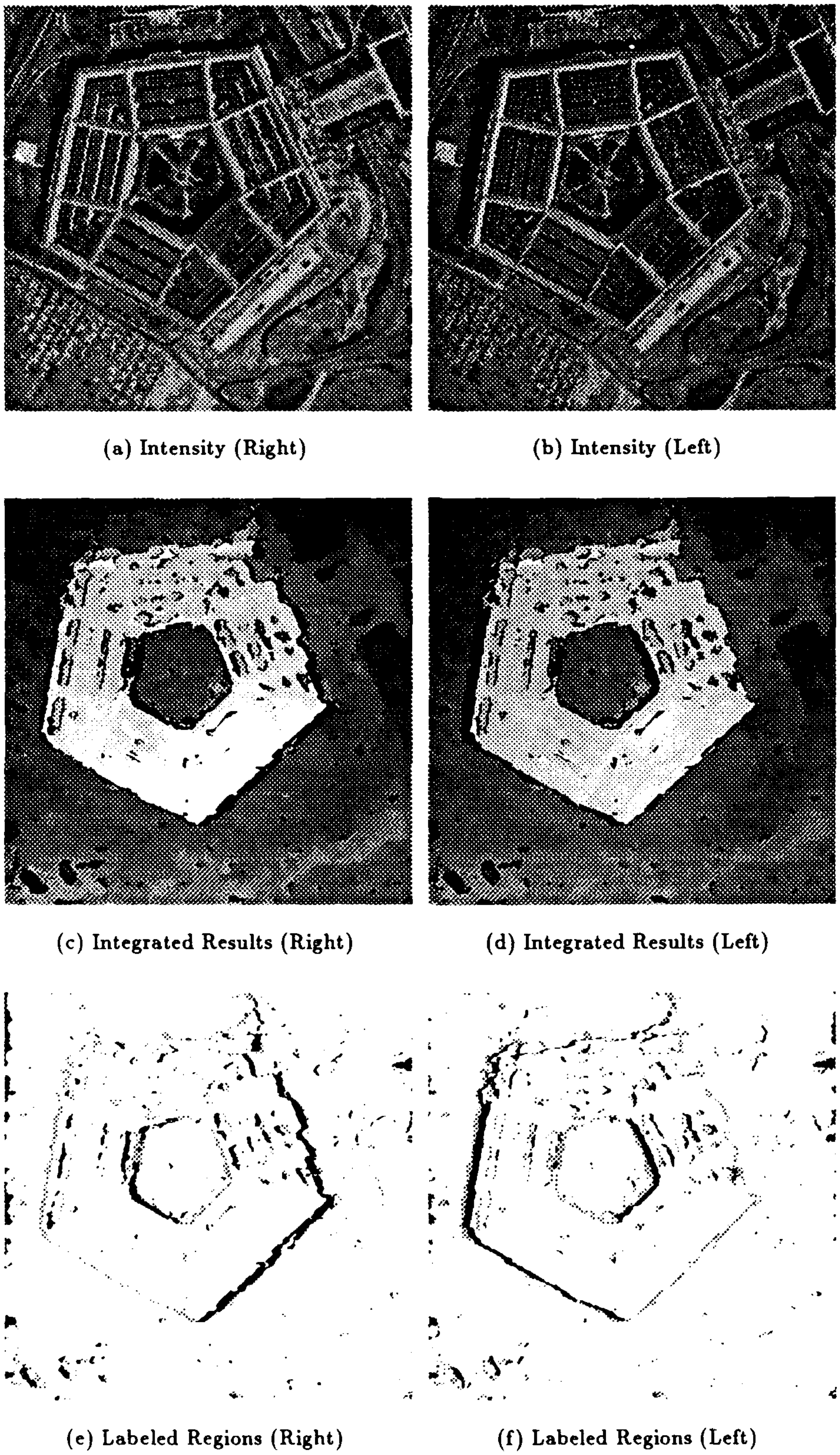


(e) 3-D Plot of the final result.



(f) 3-D Rendering of the final result.

Figure 9: (cont.) Renault Part - Reconstructions



(a) Intensity (Right)

(b) Intensity (Left)

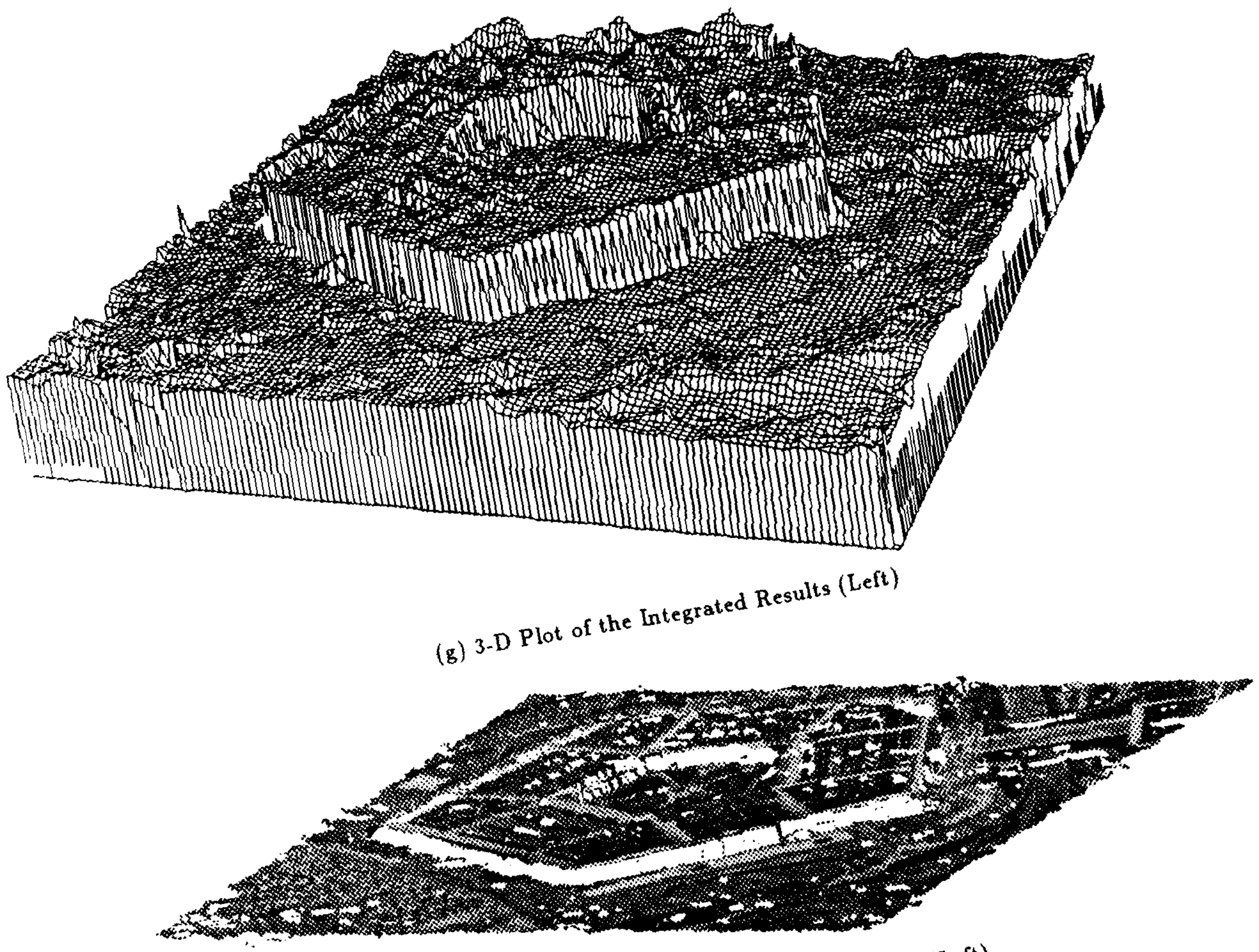
(c) Integrated Results (Right)

(d) Integrated Results (Left)

(e) Labeled Regions (Right)

(f) Labeled Regions (Left)

Figure 10: Pentagon – Intensity and Results



(g) 3-D Plot of the Integrated Results (Left)

(h) 3D Rendering of the Integrated Results (Left)

Figure 10: (cont.) Pentagon – Intensity and Results

Acknowledgment

We would like to thank Dr. Philippe Saint-Marc at the USC Institute for Robotics and Intelligent Systems for the use of his software for adaptive image smoothing and for the shading and rendering of the final results.

References

[1] S. Barnard and M. Fischler. Computational stereo. *ACM Computing Surveys*, 14(4):553–572, December 1982.

[2] K. Mori, M. Kidode, and Asada H. An iterative prediction and correction method for automatic stereocomparison. *Computer Graphics anJ Image Processing*, 2:393–401, 1973.

[3] M. Hannah. *Computer Matching of Areas in Stereo Images*. PhD thesis, Stanford University Computer Science Department, Stanford, California, July 1974. Technical Report STAN-CS-74-438.

[4] M. Hannah. SRI's baseline stereo system. In *Proceedings of the DARPA Image Understanding Workshop*, pages 149–155, December 1985. Miami Beach, Florida.

[5] M. Hannah. Bootstrap stereo. In *Proceedings of the DARPA Image Understanding Workshop*, pages 201–208, April 1980. College Park, Maryland.

[6] R. D. Arnold. *Automated Stereo Perception*. PhD thesis, Stanford University, Stanford, California, March 1983. Technical Report STAN-CS-83-961.

[7] R. D. Arnold. Local context in matching edges for stereo vision. In *Proceedings of the DARPA Image Understanding Workshop*, pages 65–72, May 1978. Boston, Massachusetts.

[8] D. Marr and T. Poggio. A theory of human stereo vision. Technical Report Memo 451, Massachusetts Institute of Technology Artificial Intelligence Laboratory, November 1977.

[9] J. E. W. Mayhew and J. P. Frisby. Psychophysical and computational studies towards a theory of human stereopsis. *Artificial Intelligence*, 17:349–385, August 1981.

[10] W. Grimson. *From Images to Surfaces: A Computational Study of the Human Early Visual System*. Artificial Intelligence. The MIT Press, Cambridge, Massachusetts, 1981. Based on the author's thesis (Ph.D. - MIT).

[11] H. Baker. Depth from edge and intensity based stereo. Technical Report STAN-CS-82-930, Stanford University, Stanford, California, September 1982. Based on the author's thesis (Ph.D. - Illinois).

[12] Y. Ohta and T. Kanade. Stereo by two-level dynamic programming. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 7(2):139–154, April 1985.

[13] G. Medioni and R. Nevatia. Segment-based stereo matching. *Computer Vision, Graphics, and Image Processing*, 31:2–18, 1985.

[14] W. Hoff and N. Ahuja. Surfaces from stereo: Integrating feature matching, disparity estimation, and contour detection. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 11(2):121–136, February 1989.

[15] T. E. Boult and L.-H. Chen. Synergistic smooth surface stereo. In *IEEE International Conference on Computer Vision*, pages 118–122, December 1988. Tampa, Florida.

[16] D. Terzopoulos. The computation of visible-surface representations. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 10(4):417–438, July 1988.

[17] A. Blake and A. Zisserman. *Visual Reconstruction*. Artificial Intelligence. The MIT Press, Cambridge, Massachusetts, 1987.

[18] P. Saint-Marc and G. Medioni. Adaptive smoothing for feature extraction. In *Proceedings of the DARPA Image Understanding Workshop*, pages 1100–1113, April 1988. Cambridge, Massachusetts.

[19] M. Drumheller and T. Poggio. On parallel stereo. In *Proceedings of the IEEE Conference on Robotics and Automation*, pages 1439–1448, April 1986. San Francisco, California.

[20] A. L. Yuille and T. Poggio. A generalized ordering constraint for stereo correspondence. Technical Report Memo 777, Massachusetts Institute of Technology Artificial Intelligence Laboratory, 1984.

[21] J. F. Canny. A computational approach to edge detection. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 8(6):679–698, November 1986.

Determination of Camera Location and Orientation

Rakesh Kumar

Computer and Information Science Dept.
University of Massachusetts at Amherst, MA. 01002.

Abstract

This paper describes a solution and mathematical analysis of the problem of estimating camera location and orientation from a set of recognized landmarks appearing in the image, sometimes refered to as pose determination. The landmarks we use are real or virtual 3D lines represented in a world coordinate system. Given correspondences between these 3D lines and 2D lines found in the image, the goal is to find the Rotation and Translation which map the world coordinate system to the camera coordinate system. The camera model assumes perspective projection.

We develop two algorithms "R_then_T" and "R_and_T" to estimate the camera location and orientation. Algorithm "R_then_T" solves for rotation first and uses the result to solve for translation. It is a variation of an algorithm developed by Liu, Huang and Faugeras [11]. The second algorithm "R_and_T" solves for both rotation and translation, simultaneously. The results from the second algorithm are much better that those of the first. We also discuss the performance of our algorithm with respect to other error measures. A closed form expression is developed for the uncertainty in the output parameters as a function of the variance of the noise in the input parameters. Based on this analysis, statements are made about the kind of errors to expect in different situations.

1 Introduction

This paper describes a solution and mathematical analysis of the problem of estimating camera location and orientation from a set of recognized landmarks appearing in the image. The landmarks employed are real or virtual 3D lines represented in a world coordinate system. Given correspondences between these 3D lines and 2D lines found in the image, the goal is to find the Rotation and Translation which map the world coordinate system to the camera coordinate system. We assume that correspondences established between model and data are line correspondences and not endpoint correspondences. The camera model assumes perspective projection. In addition, intrinsic camera parameters, such as focal length, field of view, center of the image, size of image etc. are assumed to be known [9,15,3].

We are interested in applying our algorithm to aid in the navigation of a robot moving in a known outdoor environment. The results which we present will be for situations where the landmarks are distant from the camera, to the order of hundreds of feet.

A mathematical analysis of an uncertainty measure is developed, which relates the variance in the output parameters to the noise present in the input parameters. For this analysis, we assume that there is no noise in the 3D model data and the only input noise is in the image data. To our knowledge this is the first paper which provides a mathematical analysis of the uncertainty in output parameters for the "camera location determination" problem.

1.1 Previous Work

The problem of "determination of camera location and orientation" has been referred to by various other names, "exterior orientation", "pose determination" or "pose refinement". We prefer the first name and will henceforth refer to it by its abbreviated form, "camera location determination". There have been many papers on camera location determination, but most assume point data is available and only a few have presented techniques for line data. Most solutions are also iterative in nature and require an initial estimate.

Fischler and Bolles [5] assume point data and solve for the "legs" of the points (the lengths of rays from the optical center of the camera to the points in 3D space). The closed form solution they present is quite complex and involves solving a quartic equation iteratively. Their technique is one of the very few which attempts to deal with the "outlier" problem, i.e. situations where gross errors are present and smoothing by least squares will not work.

"This research was supported by the following Defense Advanced Research Projects Agency grants F30602-87-C-0140, DACA76-85-C-0008, DACA76-86-C-0015 and National Science Foundation grant DCR8500332.

Recently, Linnainmaa et. al. have come up with a generalized hough transform approach to find the coordinates of the 3D points in camera coordinates [10].

Lowe [12] presents iterative techniques for both point and line data. However, he does not assume perspective projection and therefore his solution is not applicable to our problem of camera location determination in outdoor scenes. His technique is similar to that of Wolf [16] appearing in the photogrammetry literature. Ganapathy [6] presents a linear closed form solution for point data. Besides solving for the rotation and translation parameters, he also solves for the center of the image and scaling along "x" and "y" directions in the image. We have an implementation of his technique and find it extremely susceptible to noise, probably due to his linear least squares minimization where he assumes all his parameters are independent when they are not. Recently, Faugeras et. al. [3] have come up with a technique to solve a similar system of equations with appropriate constraints. Here it is important to draw the distinction between techniques for "camera calibration" [9,15,3], also called "interior orientation" versus the techniques for "camera location determination". Camera calibration techniques solve for intrinsic camera parameters along with the rotation and translation. The techniques for camera calibration require very precise image measurements and are less tolerant to noise. Camera location determination techniques are less susceptible to image noise but one needs to know the intrinsic camera parameters very accurately.

Liu, Huang and Faugeras present a solution to the "camera location determination" problem which works for both point and line data [11]. Our work is based on the constraints formulated by them. Their constraint uses the fact that the 3D lines in the camera coordinate system must lie on the projection plane formed by the corresponding image line and the optical center. Using this fact, constraints for rotation can be separated from those of translation. They first solve for the rotation and then use the rotation result to solve for the translation. They suggest two methods to solve for the rotation constraint. In the first method, they represent the rotation as an orthonormal matrix and devise an eigenvalue solution. However, they do not enforce the six orthonormality constraints for an orthonormal matrix. It is not clear how they would find the nearest orthonormal matrix to the matrix their algorithm returns, and whether they then would have a solution to the earlier problem. The second method represents rotation by Euler angles, and is a non-linear iterative solution obtained by linearizing the problem about the current estimate of the output parameters. The translation constraint is solved by a linear least-squares method.

1.2 Our approach

One of the main results of this paper is that the decomposition of the solution into the two stages of solving first for rotation and then translation does not use the set of constraints effectively. This same observation was made by other researchers working in the structure from motion problem [2]. The rotation and translation constraints, when used separately, are very weak constraints. When solving for them separately, even small errors in the rotation stage get amplified to large errors in the translation stage. This is particularly true with the large distances of the landmarks from the camera in our application. If we solve for them simultaneoulsy, we get much better noise immunity.

We use the same constraints as Liu, Huang and Faugeras, but a different non-linear technique. The technique we use was adapted from one used by Horn [8] to solve the problem of relative orientation (similar to structure from motion). We believe that the application of Horn's technique gives us much better convergence properties than their solution using Euler angles. With Horn's technique an implementation has been developed where a solution for rotation is obtained first, and then is used to solve for translation. We call this algorithm "R_then_T". Again using Horn's technique, another algorithm "R_and_T" was developed to solve for the rotation and translation simultaneously. We also discuss using other error functions based on similar constraints for minimization. Algorithm "R_and_T" gives the best performance in all cases.

Liu, Huang and Faugeras extend their technique to point data by drawing virtual lines between pairs of points [11]. They use the same rotational constraint; however, the translation constraint is different from that of lines. The techniques and mathematical analysis developed here apply equally well to 3D/2D point data. We have developed algorithms which use point data. These algorithms can be extended to deal with combinations of point and line data. To limit the length of the paper, only the analysis and results for the line data will be presented here. We will make the following comments about using point data. Firstly for points too, we find that solving rotation and translation simultaneously instead of separately (as they propose) gives much better results. Another observation we make is that a point algorithm using "n" points seems to be more robust than a line algorithm using "n" lines. The results for both points and lines depends on the particular data set one has. The point algorithm returns better results chiefly because the results of the first stage, i.e. the rotation stage, are much better. Intuitively, this is because using "n" points we can draw $O(n^2)$ lines.

1.3 Minimum number of lines

Both rotation and translation in the 3D world can be represented by three parameters each. Each line or point data gives us 2 constraints. Thus, a minimum of three lines or points are needed. However, in many cases, with three lines or points, there is no unique solution. If the three lines are parallel in 3D space or lie on the same projection plane, then an infinite number of solutions can be found. If the three lines meet at a common point in 3D space, then we can get two solutions for rotation (the Necker cube phenomena) and an infinite number of solutions for translation.

When the 3D lines or points lie on a plane there are always at least two global minimas (solutions) and two local minimas. The second solution can be got by reflecting the camera about the 3D plane and rotating it by 180 deg. about its axis. Fischler and Bolles [5] provide another geometric construction, where there could be up to four solutions for 3 points or lines in a plane. The same construction can be used to demonstrate more than one solution for cases of four and five points. Given a solution, they demonstrate that another solution can be constructed if the two normals drawn to a side of the triangle (formed by the 3 points or lines) from the optical center and the opposite vertex meet at a common point on the side. In this manner, for each side, we could possibly construct another solution therefore getting a maximum of four possible solutions. In general for 3D plane data, there could be a maximum of 8 solutions.

2 Rotation and Translation Constraints

The rigid body transformation from the world coordinate system to the camera coordinate system can be represented as a rotation (R) followed by a translation (T). The i'th point p_i in world coordinates gets mapped to the point p_{ci} in camera coordinates. Lines in 3D can be represented by two points p_i^R and p_i^L or a point p_i and a direction d_i. The mapping is represented by the following equation.

$$p_{ci} = R(p_i) - T \tag{1}$$

In the above equation, except for the rotation R, all the other terms are column vectors with 3 components each. We refer to the components by the subscripts x, y and z. R is the rotation operator and can be expressed in many ways, e.g. orthonormal matrices, quarternions, axis and angle, etc. Fig. 1 shows the camera and world coordinate systems. X_w, Y_w and Z_w represent the axes of the world coordinate system. O is the optical center of the lens and the origin of the camera coordinate system $OX_cY_cZ_c$. OZ_c is the optical axis. The 3D line "AB" projects to the image line "ab". A 3D point p_{ci} projects to an image pixel I_i by the following equations:

$$I_{ix} = s_x p_{cix}/p_{ciz} \qquad I_{iy} = s_y p_{ciy}/p_{ciz} \tag{2}$$

where s_x and s_y are the scale factors along the "X" and "Y" directions respectively. They are related to the field of view angles ϕ_x, ϕ_y and the image size N_r (number of rows or columns, assuming a square image) :

$$s_x = (N_r/2)\cot(\phi_x/2) \quad s_y = (N_r/2)\cot(\phi_y/2)$$

A line in the image plane can be represented in (ρ, θ) parameters by the following equation :

$$I_{ix}\cos\theta_i + I_{iy}\sin\theta_i = \rho_i \tag{3}$$

Substiuting I_{ix} and I_{iy} from equation (2) into equation (3) we get the equation of the projection plane formed by the image line and the optical center :

$$(s_x \cos\theta_i)p_{cix} + (s_y \sin\theta_i)p_{ciy} - \rho_i p_{ciz} = 0 \tag{4}$$

In Fig. 1, the projection plane formed by the image line "ab" is given by the plane "Oab" and the 3D line "AB" must lie in this plane. "N" is the normal to the projection plane, given by the vector N_i :

$$N_i = (s_x \cos\theta_i, s_y \sin\theta_i, -\rho_i)^T \tag{5}$$

N_i can be normalized to be a unit vector and henceforth we shall assume that N_i is the unit normal vector to the projection plane.

The rotation constraint for lines, formulated by Liu, Huang and Faugeras [11], that the 3D line must lie in the projection plane formed by its corresponding image line is :

$$N_i \cdot R(d_i) = 0 \tag{6}$$

We noted above that a rigid body transformation can be represented as a rotation followed by a translation. The translation does not change the direction of the line. Therefore, the direction of the 3D line after rotation must be perpendicular to the normal of the projection plane of the image line.

The translation constraint formulated by Liu et. al. uses the fact that any point on the 3D line in camera coordinates must lie on the projection plane. The vector formed from the origin (optical center) to this point must be perpendicular to the normal of the projection plane. Note, we can choose any point p_i on the 3D line. This can be expressed as follows :

$$N_i \cdot (R(p_i) - T) = 0 \quad or \quad N_i \cdot T = -N_i \cdot R(p_i) \tag{7}$$

At this point, we would like to make clear the two algorithms we have developed and will be comparing in this paper. In the first algorithm "R_then_T" we solve for rotation using the constraint in equation (6). Then, the rotation result returned from this step, is used in conjunction with equation (7) to solve for translation.

In the second algorithm "R_and_T", only equation (7) is used to solve for both rotation and translation simultaneously. For each line, two points are used which must satisfy equation (7). As the tables in our results section will show, "R_and_T" performs much better then "R_then_T".

3 Least Square Solution methods

Ideally we would like to find the rotation "R" and translation "T" by which equation (7) is satisfied for each line. With noise, however, this will not be possible. In the "R_and_T" case, the objective function "E_1" we minimize is given by :

$$E_1 = \sum_{i=1}^{2n} (N_i \cdot (R(p_i) + T))^2 \tag{8}$$

For each image line two 3D points are used and therefore, each line contributes twice to the objective function. A physical interpretation of the objective function above is that it is the sum-of-the-squares of the perpendicular distances from the end-points of the 3D lines to their corresponding projection planes formed using the image lines. We minmize "E_1" to find an "R" and "T" such that the end-points of the 3D lines are mapped as close as possible (in terms of sum of squares of perpendicular distances) to their corresponding projection planes.

The above objective function E_1 gives higher weighting to endpoints of 3D lines which are further away from the camera. In order to give equal weighting to all endpoints, the following objective function is to be minimized :

$$E_2 = \sum_{i=1}^{2n} \left(\frac{N_i \cdot (R(p_i) + T)}{\mid R(p_i) + T \mid}\right)^2 \tag{9}$$

E_2, unlike E_1, is a rational function and therefore more difficult to optimize. To minimize E_2, at each iteration in our non-linear technique, we hold the denominator $\mid R(p_i) + T \mid$ for each point to be constant. In the next iteration, the value of $\mid R(p_i) + T \mid$ is updated with the new "R" and "T". Therefore, we are able to employ the same algorithm as used for E_1. This seems to work for all the cases we have run our algorithm on.

In contrast to E_1 and E_2, we could construct an objective function E_3, which minimizes the sum-of-the-squares of the perpendicular distances of the end-points of the image lines to the projection plane formed using the 3D line and the optical center.

$$E_3 = \sum_{i=1}^{n} \sum_{j=1}^{2} \left(\frac{p_{i1} - T_w}{\mid p_{i1} - T_w \mid} \times \frac{p_{i2} - T_w}{\mid p_{i2} - T_w \mid} \cdot (R^T I_{ij})\right)^2 \tag{10}$$

where T_w and R^T are the translation and rotation in the world coordinate system. We minimize E_3 by a method similar to the techniques used for E_1 and E_2. We found the performance of algorithm optimizing E_3 to be poorer, with respect to noise, than algorithms minimizing E_1 and E_2. We suspect this is because the numerator of E_3 is a fourth order function of "R" and "T" as compared to the second order numerators of E_1 and E_2.

Finally, in the "R_then_T" case, the rotation objective function E_R and translation objective function E_T are :

$$E_R = \sum_{i=1}^{n} (N_i \cdot R(d_i))^2 \qquad E_T = \sum_{i=1}^{n} (N_i \cdot (R(p_i) + T))^2 \tag{11}$$

We first find the "R" that minimizes E_R and then use that "R" to find a "T" which minimizes E_T.

E_1, E_2, E_3 and E_R are all minimized by modifiying the same basic non-linear technique. Therefore only the technique for minimizing E_1 is presented. E_T is minimized by a straight-forward linear least squares.

3.1 Non-linear Technique for "R_and_T"

To minimize "E_1", we adapt an iterative technique formulated by Horn [8] to solve the problem of Relative Orientation. It needs an initial estimate for both "R" and "T". The technique linearizes the error term about the current estimate for "R" and "T". It is then possible to determine how the overall error is affected by small changes in rotation and translation. This allows one to make iterative adjustments to the rotation and translation that reduce "E_1". These iterations are continued until the algorithm converges to a minimum. Note that the algorithm, like all such descent algorithms, does not guarantee a global minimum.

Assume we have a current estimate "R" for rotation. The coordinates p'_i of a rotated 3D point is given by $p'_i = R(p_i)$. We add an incremental rotation vector $\delta\omega$ to the rotation estimate "R". The direction of this incremental vector is parallel to the axis of rotation, while its magnitude is the angle of rotation. This incremental rotation takes p'_i to p''_i

$$p''_i = p'_i + \delta\omega \times p'_i \tag{12}$$

This follows from Rodrigue's formula [8] for the rotation of a vector "r" by angle "θ" about axis "ω" :

$$r' = r(cos\theta) + sin\theta(\omega \times r) + (1 - cos\theta)(\omega \cdot r)\omega \tag{13}$$

where $\theta = ||\delta\omega||$ and $\delta\omega = \delta\omega/||\delta\omega||$

Let ΔT represent a small translation added to the current translation estimate T. Thus, the linearized energy function about the current estimate "R" and "T" becomes :

$$E = \sum_{i=1}^{2n} (N_i \cdot (p'_i + \delta\omega \times p'_i + T + \Delta T))^2 \tag{14}$$

Let $b_i = p'_i \times N_i$. Using the chain rule of triple scalar product for vectors, differentiating the objective function with respect to ΔT and $\delta\omega$ respectively, and setting the results equal to 0, after some manipulation the following two equations are obtained :

$$\sum_{i=1}^{2n} (N_i \cdot \Delta T + \delta\omega \cdot b_i)N_i = -\sum_{i=1}^{2n} (N_i \cdot (p'_i + T))N_i \tag{15}$$

$$\sum_{i=1}^{2n} (N_i \cdot \Delta T + \delta\omega \cdot b_i)b_i = -\sum_{i=1}^{2n} (N_i \cdot (p'_i + T))b_i \tag{16}$$

Together, the above two vector equations constitute 6 linear scalar equations in the 6 unknown components of ΔT and $\delta\omega$. We can rewrite them in the more compact matrix form:

$$\begin{aligned} C\Delta T + F\delta\omega &= -\bar{c} \\ F^T\Delta T + D\delta\omega &= -\bar{d} \end{aligned} \tag{17}$$

where $C = \sum_{i=1}^{2n} N_i N_i^T \quad D = \sum_{i=1}^{2n} b_i b_i^T \quad F = \sum_{i=1}^{2n} N_i b_i^T$
while $\bar{c} = \sum_{i=1}^{2n} (N_i \cdot (p'_i + T))N_i$ and $\bar{d} = \sum_{i=1}^{2n} (N_i \cdot (p'_i + T))b_i$.

Solving the above set of 6 linear equations gives a way of finding small changes in rotation and translation that reduce the overall objective function. The algorithm can therefore be expressed in the following four steps.

Step 1 Guess an initial estimate for rotation "R" and translation "T".

Step 2 Compute the coefficients of the matrices in equation (17). Solve the linear system for ΔT and $\delta\omega$.

Step 3 Compose $\delta\omega$ with the current estimate R of rotation to get the new estimate. Add ΔT to T to get the next estimate for translation.

Step 4 If the algorithm has converged or has exceeded a maximum number of iterations, else go back to Step 2.

We compose the $\delta\omega$ and ΔT to the current estimate and iterate. We stop iterating when either a maximum number of iterations is exceeded or when the difference in the result between two successive iterations is less than a prespecified minimum. The algorithm seems to converge for initial estimates which differ considerably from the correct solution. Incremental adjustments cannot be computed if the six-by-six coefficient matrix becomes singular. This will happen when we have a situation for which there are infinite solutions. To compose the $\delta\omega$ with the current estimate of the Rotation R, the current rotation is represented as a quarternion [8].

4 Uncertainty Analysis

Noise is assumed to be in the image data only; and the 3D model data is assumed to accurate. In this section closed form expressions are developed for the variance of the error in the output parameters (rotation and translation) as a function of the input data and variance of the noise and the output translation and rotation values. In the analysis, as in all such statistical analyses [13,1], the basic assumption is that the returned output parameter is the true or correct output parameter and the uncertainity region is centered around it. The analysis is local in nature. We assume our solution is at the global minimum and near the true solution. This condition could be violated by our algorithm if the initial estimate was poorly chosen. In our domain of robot navigation using landmarks, this is unlikely to happen because the "camera location determination" step is used to refine the position of the robot, which has moved a few feet from its previous known position.

The image data for lines can be specified by two parameters ρ_i and θ_i as in equation (3). For the analysis, we assume the noise for both ρ_i and θ_i is Gaussian distributed, zero mean and uncorrelated with variance $\sigma^2_{\rho_i}$ and $\sigma^2_{\theta_i}$. respectively. Instead of assuming zero-mean gaussian noise for the (ρ_i, θ_i) parameters of the noise, the assumption could be made that the endpoints of the line are zero-mean gaussian. The following derivation can be easily modified for that case.

The error in translation ΔT and rotation $\delta\omega$ is expressed as a function of the input data, output translation and rotation values and input noise. The objective function is linearized around the computed Rotation"R" and Translation "T". Minimizing the linearized energy function enables us to express ΔT and $\delta\omega$ as linear functions of Δ_{ρ_i} and Δ_{θ_i}, the error in the input data.

The variances of ΔT and $\delta\omega$ are computed using the linear functions. We also compute the expected value of the objective function. Finally, we check if the linearization is valid by determining whether or not if the actual objective function value is of the order of the computed expected value (as predicted by linearizing). If not, we disregard the output variances we have computed.

4.1 Error expression for normals of the projection planes

The image data in the rotation and translation constraints appears in the form of the normals of the projection planes formed by the image lines and the focal point. Therefore, we represent small errors in the ρ, θ specifications of the image lines as errors in the normals of the projection plane. The normal vector is given in equation (5). Let us consider the errors in ρ, θ to be small and given by $\Delta\rho, \Delta\theta$ respectively. Equation (5) then becomes :

$$N'_i = (s_x \cos(\theta_i - \Delta\theta_i), s_y \sin(\theta_i + \Delta\theta_i), -\rho_i - \Delta\rho_i) \qquad (18)$$

N'_i is to be normalized. After normalizing the error in the normal vector, ΔN_i can be expressed as follows :

$$\Delta N_i = 1/M_i(-s_x \sin(\theta_i)\Delta\theta_i, s_y \cos(\theta_i)\Delta\theta_i, -\Delta\rho_i) \qquad (19)$$

where "M_i" is the magnitude of N'_i. We approximate "M_i" by :

$$M_i \approx ((s_x \cos\theta_i)^2 + (s_y \sin\theta_i)^2 + \rho_i^2)^{1/2} \qquad (20)$$

Two observations can be made :

1. The components of ΔN_i are scaled by M_i. One of the terms in M_i is the square of ρ_i. The larger the ρ_i of a line, the smaller the components of ΔN_i will be. Thus, our algorithms would be more tolerant to noise in lines, which have a larger ρ_i. The effects of this are not too significant, because the sum of the other terms in M_i will be larger than the square of ρ_i. Nevertheless, it is an outcome of forming normals of projection planes from image lines.
2. Rotation of image lines due to noise is more harmful than translation of image lines. The rotation terms involving $\Delta\theta$ are scaled by s_x and s_y. The translation term $\Delta\rho$ will be generally smaller. This is borne out by our experiments, as can be seen in Table 1 in the result section for the 5 line case.

4.2 Variance in output parameters for Algorithm R_and_T

Our algorithm tries to minimize the energy term E_1 given by equation (8). Let us assume that N_i, R, T and p_i are the correct or true normals, rotation, translation and 3D points respectively. If we subsitute them into equation (8),

E_1 should exactly be equal to zero. Now, we add noise ΔN_i *to the normals* N_i . *We wish to find the expressions* which relate the noise in the output parameters, $\delta\omega$ for rotation and ΔT for translation to the input noise. We assume the error, at least for rotation, is small, that is, less than 20 deg. around each axis. The energy term E_1 can be rewritten in terms of ΔN_i, $\delta\omega$ and ΔT. N_i, R, T and p_i are assumed to be constant and represent the true values. Given ΔN_i, we can solve for $\delta\omega$ and ΔT by minimizing the new energy term E'. Based on equation (12) and (14) the energy term E' can be rewritten as :

$$E' = \sum_{i=1}^{2n}((N_i + \Delta N_i) \cdot (p_i' + \delta\omega \times p_i' + T + \Delta T))^2 \tag{21}$$

where, as before $p_i' = RPp_i$. Now, from our above assumptions $N_i \cdot (p_i' + T) = 0$. Therefore ignoring second order terms, E' now becomes:

$$E' \approx \sum_{i=1}^{2n}(\Delta N_i \cdot (p_i' + T) + N_i \cdot (\delta\omega \times p_i' + \Delta T))^2 \tag{22}$$

Ignoring the second order terms in (21) basically means assuming that $N_i \cdot (\delta\omega \times p_i' + \Delta T)$ is approximately equal to $(N_i + \Delta N_i) \cdot (\delta\omega \times p_i' + \Delta T)$.

After differentiating E' in the above equation with $\delta\omega$ and ΔT respectively and setting the result equal to zero, the following two functions for ΔT and $\delta\omega$ are obtained :

$$\Delta T = -G1\hat{c} - G2\hat{d} \tag{23}$$

$$\delta\omega = -G2^T\hat{c} - G4\hat{d} \tag{24}$$

where $G1 = (C^{-1} - C^{-1}FG2^T)$ $\quad G2 = -(C^{-1}FG4)$ $\quad G4 = (D - F^TC^{-1}F)^{-1}$
$\hat{c} = \sum_{i=1}^{2n}(\Delta N_i \cdot p_{ci})N_i$ $\quad \hat{d} = \sum_{i=1}^{2n}(\Delta N_i \cdot p_{ci})b_i$
$C = \sum_{i=1}^{2n} N_i^T N_i$ $\quad D = \sum_{i=1}^{2n} b_i^T b_i$ $\quad F = \sum_{i=1}^{2n} N_i^T b_i$
$b_i = p_i' \times N_i$ $\quad p_{ci} = R(p_i + T.\ p_{ci}$

We introduce two more symbols U_i and V_i :

$$U_i = G1_{i1}N_{ix} + G1_{i2}N_{iy} + G1_{i3}N_{iz} + G2_{i1}b_{ix} + G2_{i2}b_{iy} + G2_{i3}b_{iz}$$
$$V_i = G2_{i1}^TN_{ix} + G2_{i2}^TN_{iy} + G2_{i3}^TN_{iz} + G4_{i1}b_{ix} + G4_{i2}b_{iy} + G4_{i3}b_{iz}$$

Using the above two expressions for U_i and V_i, equation (24) and the expansions for $\hat{c}$ and $\hat{d}$ we can write expressions for ΔT and $\delta\omega$ in terms of $\Delta\rho_i$ and $\Delta\theta_i$.

$$\Delta T_i = \sum_{i=1}^{2n}(\frac{U_i}{M_i}(p_{cix}s_x \sin\theta_i - p_{ciy}s_y \cos\theta_i)\Delta\theta_i + p_{ciz}\Delta\rho_i)) \tag{25}$$

$$\delta\omega_i = \sum_{i=1}^{2n}(\frac{V_i}{M_i}(p_{cix}s_x \sin\theta_i - p_{ciy}s_y \cos\theta_i)\Delta\theta_i + p_{ciz}\Delta\rho_i) \tag{26}$$

From statistics [1], we know that the variance of a parameter "z" which is a function of input parameters "y_i" is given by

$$\sigma_z^2 = \sum \left[\sigma_{y_i}^2 \frac{\delta z}{\delta y_i}\right] \tag{27}$$

Note, that each line will contribute two terms, one for each point, to the summation in the two equations (25,26). Let us denote the two points we use for a line to be p_i^R and p_i^L. For both these 3D points, the corresponding projection plane normals would be the same and so would ΔT_i and $\delta\omega_i$. Using the above equations (25), (26) and (27) we can write the following closed form expressions for the variance in the output parameters.

$$\begin{aligned}\sigma_{\Delta Ti}^2 \quad &= \textstyle\sum_{i=1}^{n}(\quad ((U_i^L p_{cix}^L + U_i^R p_{cix}^R)s_x \sin\theta_i - (U_i^L p_{ciy}^L + U_i^R p_{ciy}^R)s_y \cos\theta_i))^2 \frac{\sigma_{\Delta\theta_i}^2}{M_i^2} + \\ &\qquad (U_i^L p_{ciz}^L + U_i^R p_{ciz}^R)^2 \frac{\sigma_{\Delta\rho_i}^2}{M_i^2})\end{aligned} \tag{28}$$

$$\sigma^2_{\delta\omega i} = \sum_{i=1}^{n}\Big(((V_i^L p_{cix}^L + V_i^R p_{cix}^R) s_x \sin\theta_i - (V_i^L p_{ciy}^L + V_i^R p_{ciy}^R) s_y \cos\theta_i))^2 \frac{\sigma^2_{\Delta\theta_i}}{M_i^2} + (V_i^L p_{ciz}^L + V_i^R p_{ciz}^R)^2 \frac{\sigma^2_{\Delta\rho_i}}{M_i^2}\Big) \tag{29}$$

Note $\sigma_{\Delta T_i}$ for i = 1..3 corresponds to $\sigma_{\Delta T_x}$, $\sigma_{\Delta T_y}$ and $\sigma_{\Delta T_z}$ respectively, and $\sigma_{\delta\omega_i}$ for i = 1..3 corresponds to $\sigma_{\delta\omega_x}$, $\sigma_{\delta\omega_y}$ and $\sigma_{\delta\omega_z}$ respectively.

From equations (28,29), it can be seen that the squares of both σ_{Ti} and $\sigma_{\delta\omega i}$ have a quadratic dependence on p_c^R and p_c^L, the location of the 3D endpoints in camera coordinates. Therefore, 3D lines which are closer to the camera, will contribute less to the error variance in the output parameters. This is to be expected, since the projections of these lines changes the most for a small change in translation. Lines parrallel to the x-axis will not constrain the translation along the x-axis or rotation about it. Similiarly, lines parrallel to the y-axis and z-axis, will not constrain the output parameters along the y-axis and z-axis, respectively.

5 Results and Discussion

The development of the algorithms presented here is part of a larger effort to have the UMASS robot "Harv" navigating the sidewalks of the UMASS campus [4]. We tested our algorithm using a 3D model of the campus environment around the Graduate Research Center. Fig. 2a and Fig. 3a are example images on which we tested our algorithm. Experiments were conducted with both real image data and simulated data with noise added to it. The landmarks used were the 3D lines forming the visible corner of the building, window lines, lampposts, telephone poles and one sidewalk line. The experiments for both synthetic and real data were conducted with the camera being about 300 feet distant from the building in Figs. 2a and 3a. The synthetic data experiments were conducted with projections of 3D lines from the model. The camera was assumed to be placed at the same location as the first frame of the real data experiments. For that frame, there was one telephone pole 50 feet away from the camera. The rest were in the range of 150 to 300 feet away.

We used a SONY B/W camera, model AVC-D1. Linked to a GOULD frame grabber, 512 by 484 size images are obtained, with field of view of 24.0 deg. by 23.0 deg.. Knowledge of the intrinsic parameters of the camera are extremely important for our real data experiments.

The 3D model was built over two passes. In the first pass, it was built using blueprints of the campus. These blueprints are drawn to a scale of 40 feet to an inch. We found errors of up to 10 feet in this 3D model. In the second pass, we surveyed the landmarks using theodolites. We believe most of our 3D model is now accurate to within 0.3 feet. Some landmarks, such as poles and posts, are difficult to position accurately, because of their cylindrical shape and no good distinguishing points. Accuracy of the 3D model is very important for our present experiments. An error of 1 foot in the location of a 3D landmark, 50 feet away from the camera, can cause it to be displaced by 24 pixels in the image.

5.1 Synthetic Data experiments

The synthetic data experiments were conducted for both algorithms "R_and_t" and "R_then_t". The two algorithms were run with four different sets of lines, each set being perturbed by atleast two different amounts of noise. Zero mean uniform noise was added to the ρ and θ of each image line. In Tables 1 and 2 the noise for each simulation is specified in the ρ and θ columns. One pixel noise in ρ means that to the ρ of each line, we added a $\Delta\rho$, which was a random number anywhere in the range [-1,+1]. Similiarly, one deg. noise in θ means that to the θ of each line, we added a $\Delta\theta$, which was a random number anywhere in the range [-1,+1]. We did our simulations for 1 deg. or 5 deg. noise in θ and 1 pixel or 5 pixel noise in ρ. For each set of lines and each specification of input noise, we created 100 data samples, by starting our random number generator each time with a different seed point. The results presented in the Tables, are the average absolute error of the computed rotation and translation over these 100 data samples, for each set of lines and each noise specification. The results for the "R_then_T" and "R_and_T" algorithm are in Table 2 and Table 1, respectively. Rotations and translation errors in the Tables for synthetic data are specified with respect to the camera coordinate system (Fig 1). The rotation errors are specified in terms of error in degrees of the axis-angle 3D rotation vector. ΔT_x corresponds to error in translation in the direction of the rows in the image plane (in camera coordinates). ΔT_y corresponds to error in translation in the vertical direction in camera coordinates. ΔT_z corresponds to error in translation along the direction of the optical axis in camera coordinates.

The first set of 5 lines consisted of the 4 corner edges of the building visible in Fig. 2 and one window line in that same building. The second set of 10 lines consisted of the 4 corner edges of the building, as above, and 6 lampposts and telephone pole lines. The third set of 14 lines consisted of these 10 lines plus three more lines on the building and one side walk line. The fourth set of 30 lines consisted of the above 14 lines. Plus, we assumed we had been able to identify 6 vertices, e.g. the corner of the building and drew virtual lines between these vertices if they were not already joined.

As. can be seen by comparing the Tables, the results of "R_and_T" algorithm in Table 1 are much better then the results of "R_then_T" algorithm in Table 2. With zero noise specified, both algorithms gave the correct result. For each set of lines and each specification of noise, "R_and_T" performs much better than "R_then_T". The results for "R_then_T" are particularly bad for the 5 and 10 line simulations. This can be explained by the observation that, in the 5 line case, 3 of the lines form a trihedral junction. As noted before, for trihedral junctions we can have an infinite number of translations. Thus, the translation is pinned from this infinite set by just the remaining two lines, both of which are vertical and not too far from each other. With noise, therefore we would expect large errors in translation. Similiarly, in the 10 line simulation, most of the lines are vertical. Vertical lines, do not disamiguate rotations about the x-axis and translations along the y-axis. This problem is even more compounded when the rotation stage is separated from the translation stage.

In the Tables for both algorithms, we notice that the error decreases appreciably, decreases as the number of lines increases. In the "R_and_T" Table, we give results of experiments with the 5 line data set for two extra cases. If we look in Table 1 at the results of the 5 line data sets, we find appreciably larger error when the noise in θ is 5 deg.. However, when the noise in ρ is 5 pixels and the noise in θ 1 deg., the errors are much smaller. This demonstrates what we had predicted via the uncertainity analysis section: noise in θ for lines is much more harmful than noise in ρ. Finally, in all experiments, the error in ΔT_y was found to be often larger than the errors in ΔT_x and ΔT_z. This is because of the fact that the majority of our 3D lines are vertical.

5.2 Real Data Results

The real data experiment used only the "R_and_T" algorithm. A sequence of 6 frames was used for this experiment. The camera was moved in an approximate translatory motion 25 feet along the walkway. Each subsequent frame was taken after a movement of 5 feet down the walkway. The sidewalk line is close to parallel to the x-axis in the world coordinate system. The z-axis is the vertical axis in the world coordinate system. The 2D images lines were taken from the output of a 2D line matching system [14], this is part of our mobile robot project. For each frame, column 2 in table 3 gives the number of lines the 2D line matcher was able to correctly match. Fig. 2a and Fig. 3a are images with the 2D lines of frame 2 and frame 4, respectively, as returned by the line matcher. These were used as input to our algorithm along with the 3D model.

Table 3 gives the estimated error of our algorithm for translation in world coordinates. In most cases, the robot is located to within a foot. The errors in table 3 are approximate to 0.5 feet. The precise location of the camera is not known. It is better to judge the performance of the algorithm by looking at the projections of the 3D landmarks on the image after the pose has been estimated. Fig. 2b and Fig. 3b are the projections of the 3D landmark lines from the position returned by algorithm "R_and_T". As mentioned earlier, the 2D data used is the output of the 2D line matcher. We get better results when the image lines are found by a sub-pixel line locator.

The results can be improved by the following four measures. (1) Use more lines, some of which can be obtained by drawing virtual lines from the corner of the building to the tops of lampposts. (2) Use closer landmarks. The most accurate result is for frame 1; this is probably, because it is the only frame in which there is a telephone pole 50 feet away. (3) We may not know all our intrinsic camera parameters accurately. Work is underway to calibrate the camera very precisely. (4) Improve the 3D positioning of the lampposts and poles.

Finally, the algorithm as it stands is not able to handle outliers. Sometimes the 2D line matcher errs and matches a wrong set of telephone lines. In the case of the telephone poles this introduces an error of 50 pixels or so. Our algorithm cannot recover from this by just looking at the residue errors for each line, from a least mean square analysis. More robust methods have to be employed to detect outliers. This is part of our current research.

Acknowledgements

A lot of this paper was made possible by discussions and debates with Harpreet Singh Sawhney, Raghavan Manmatha, Brian Burns, John Oliensis, Rich Weiss, Chris Connolly and Ross Beveridge. My advisors Ed Riseman and Al Hanson were very supportive of the work, provided challenging ideas and allowed me to work in a stimulating environment. Renee Stephens

helped in making the figures. Matt Easely helped in collecting the data.

References

[1] P. R. Bevington, *Data Reduction and Error Analysis for the Physical Sciences*, McGraw-Hill, NY, 1969.

[2] R.Manmatha,R.Dutta,E.M. Riseman and M.Snyder, "Issues in Extracting Motion Parameters and Depth from Approximate Translational Motion", *IEEE Workshop on Visual Motion - PRoceedings*, March 1989, pgs 264-272.

[3] O.D. faugeras and G.Toscani, "Camera Calibration for 3D Computer Vision", *Proc. International Workshop on Machine Vision and Machine Intelligence*, Tokyo,Japan, FEb 2-5,1987.

[4] C. Fennema, A. Hanson, and E. Riseman, "Towards Autonomous Mobile Robot Navigation", to appear *Proc. DARPA Image Understanding Workshop*, Morgan Kaufman Publishers, Palo Alto, CA, May 1989.

[5] M. A. Fischler ard R. C. Bolles, "Random sample consensus: A paradigm for model fitting with applications to image analysis and automated cartography," *Commun. ACM*, vol. 24, pp. 381-395, 1981.

[6] S. Ganapathy, "Decomposition of transformation matrices for robot vision," in *Proc. 1st IEEE Conf. Robotics*, pp. 130-139, 1984.

[7] B. K. P. Horn, "Closed-form solution of absolute orientation using unit quarternions," *J. Opt. Soc. A.* vol. 4, pp. 629-642, 1987.

[8] B. K. P. Horn, "Relative Orientation," *Proceedings: Image Understanding Workshop*, vol. 2, pp. 826-837, 1988.

[9] R.K. Lenz and R.Y.Tsai, "Techniques for calibiration of the scale factor and image center for high accuracy 3-D machine vision metrology," *IEEE Trans. Pattern Anal. Machine Intell.*, vol. 10 # 5, pp. 713-719, 1988.

[10] S. Linnainmaa, D. Harwood and L.S. Davis, "Pose determination of a three-dimensional object using triangle pairs," *IEEE Trans. Pattern Anal. Machine Intell.*, vol. 10 # 5, pp. 634-647, 1988.

[11] Y. Liu, T. S. Huang and O. D. Faugeras, "Determination of camera location from 2D to 3D line and point correspondences," *IEEE Int. Conf. Computer Vision and Pattern Recognition*, pp. 82-88, 1986.

[12] D. G. Lowe, *Perceptual Organization and Visual Recognition*, Kluwer Academic Publishers, Hingham, MA, 1985.

[13] W. H. Press, B. P. Flannery, S. A. Teukolsky, W. T. Vetterling, *Numerical Recipies*, Cambridge University Press, Cambridge, MA, 1986.

[14] J. Ross Beveridge, R. Weiss and E. Riseman, "Optimization of 2-Dimensional Model Matching", to appear *Proc. DARPA Image Understanding Workshop*, Morgan Kaufman Publishers, Palo Alto, CA, May 1989.

[15] R. Y. Tsai, "An Efficient and Accurate Camera Caliberation Technique for 3D Machine Vision," *IEEE Int. Conf. Computer Vision and Pattern Recognition*, pp. 364-374, 1986.

[16] P. R. Wolf, *Elements of Photogrammetry*, McGraw Hill, New York, 1974.

Table 1: **Average Absolute Error of Translation and Rotation for algorithm "R_and_T"** The average for each experiment is taken over 100 samples of uniform noise.

NOISE			ROTATION ERROR			TRANSLATION ERROR		
No. Lines	θ deg.	ρ pixels	$\delta\omega_x$ deg.	$\delta\omega_y$ deg.	$\delta\omega_z$ deg.	ΔT_x feet	ΔT_y feet	ΔT_z feet
Correct			0.00	0.00	0.00	0.00	0.00	0.00
5	1.0	1.0	0.24	0.15	0.04	0.21	2.03	1.16
5	5.0	5.0	1.20	0.79	0.19	1.08	10.14	6.20
5	1.0	5.0	0.24	0.16	0.04	0.21	2.04	1.18
5	5.0	1.0	1.19	0.78	0.19	1.08	10.14	6.20
10	1.0	1.0	0.21	0.08	0.05	0.02	1.73	0.08
10	5.0	5.0	0.72	0.27	0.31	0.18	6.33	0.48
14	1.0	1.0	0.07	0.06	0.08	0.03	0.77	0.02
14	5.0	5.0	0.34	0.30	0.39	0.17	3.80	0.12
30	1.0	1.0	0.03	0.05	0.06	0.06	0.48	0.06
30	5.0	5.0	0.16	0.24	0.31	0.32	2.39	0.32

Table 2: **Average Absolute Error of Translation and Rotation for algorithm "R_then_T"** The average for each experiment is taken over 100 samples of uniform noise.

NOISE			ROTATION ERROR			TRANSLATION ERROR		
No. Lines	θ deg.	ρ pixels	$\delta\omega_x$ deg.	$\delta\omega_y$ deg.	$\delta\omega_z$ deg.	ΔT_x feet	ΔT_y feet	ΔT_z feet
Correct			0.00	0.00	0.00	0.00	0.00	0.00
5	1.0	1.0	1.08	5.06	0.62	11.44	13.96	51.16
5	5.0	5.0	3.19	14.65	1.62	32.69	39.85	149.40
10	1.0	1.0	0.50	2.26	0.31	9.24	8.83	8.84
10	5.0	5.0	2.44	10.45	1.28	40.83	40.03	38.65
14	1.0	1.0	0.29	0.29	0.18	0.35	2.37	0.23
14	5.0	5.0	1.50	1.56	0.91	1.92	12.44	1.27
30	1.0	1.0	0.09	0.10	0.13	0.40	1.01	0.36
30	5.0	5.0	0.45	0.50	0.66	2.09	5.05	1.82

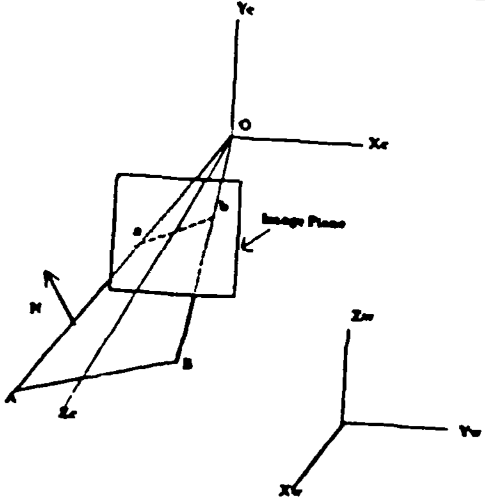


Figure 1. Camera and world coordinate system (perspective projection).

Table 3: Real Data results for a sequence of 6 frames. Estimated Errors of Translation for "R and T"

Frame No.	Num. Lines	TRANSLATION ERROR		
		ΔT_x feet	ΔT_y feet	ΔT_z feet
1	17	0.10	0.06	-0.03
2	15	0.38	-0.25	0.13
3	12	-1.1	0.3	0.10
4	7	0.57	0.86	0.65
5	13	1.60	0.72	0.54
6	13	1.87	1.10	0.72

Fig 2a. Input lines from 2D Line matcher for Frame 2.

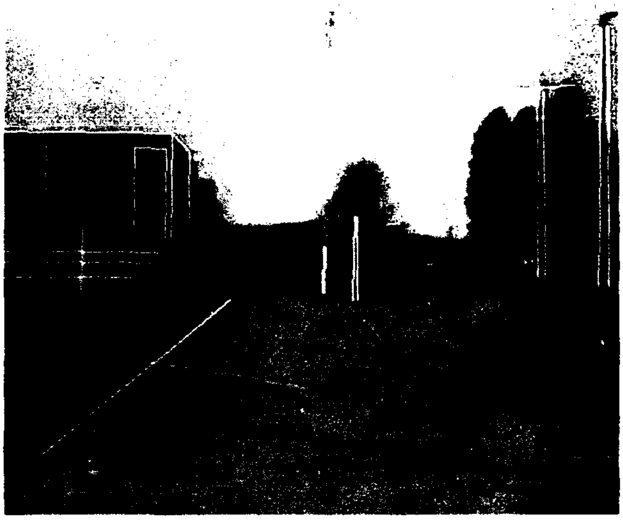


Fig 2b. Projected Lines after estimation of pose for Frame 2.

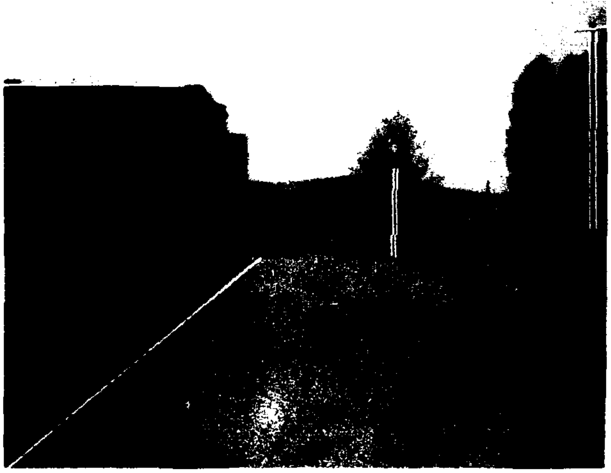


Fig 3a. Input lines from 2D Line matcher for Frame 4.

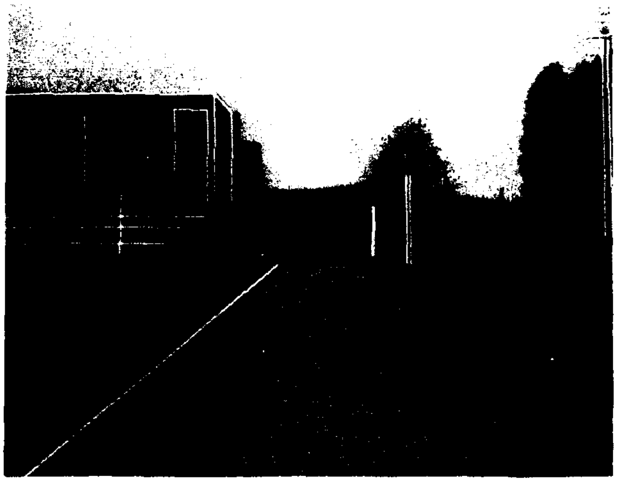


Fig 3b. Projected Lines after estimation of pose for Frame 4.

CONSTRAINTS FOR INTERPRETATION OF PERSPECTIVE IMAGES *

Fatih Ulupınar and Ramakant Nevatia
Institute for Robotics and Intelligent Systems
University of Southern California
Los Angeles, California 90089-0273

ABSTRACT

Problem of surface orientation recovery from line drawings in a single image, obtained under perspective projection is studied. Two constraints, shared boundary constraint and the orthogonality constraint previously used in orthographic projection are extended to perspective projection. New constraints are derived from observations of parallelism and a new kind of symmetry that we define, called the convergent symmetry. Convergent symmetry is the type of symmetry we get when we project a symmetric object under perspective projection. Unlike skew symmetry, convergent symmetry provides sufficient constraints to recover unique surface orientations. The set of techniques given should allow extension of all previous orthographic analysis and provides new tools for additional, more constrained analysis. An example illustrating the use of our techniques is provided. Finally, extension of the constraints for some class of curved surfaces is discussed.

1 INTRODUCTION

Inferring the 3-d shape of an object by using only the contours of its image has be a problem of prime interest in computer vision. The problem is, of course, inherently ambiguous as many objects can give rise to the same contours under different imaging conditions. However, some constraints on the 3-d positions and orientations do follow from the image, and even unique answers can be found in some cases if certain regularity assumptions can be made. This paper provides some new constraints that apply in the case of perspective projection.

The problem of line interpretation was first studied in depth for scenes of polyhedra. Pioneering contributions were made by Huffman and Clowes [Huf71] [Clo71] in rules for line labeling. Mackworth [Mac73], based partly on work of Huffman, derived some quantitative constraints on orientations of the planes in the line drawings. Kanade [Kan81] showed how incorporating some regularity constraints, particularly symmetry constraints, allowed for unique interpretations in some cases. More recently Sugihara [Sug86] has developed a comprehensive set of techniques to analyze polyhedral scenes.

Several researchers have also attempted to develop methods that apply to non-polygonal and non-planar curves and surfaces also [BY84] [BT81] [Wei88] [Asa87] [Ste81] [TX87] [UN88]. In [UN88] we provide a comprehensive review of these techniques and some new techniques for analysis of a broad class of curved surface scenes.

Usually, such analysis assumes *orthographic* projection. In this paper, we derive constraints that apply to the more general *perspective* projection. It has been conventionally thought that constraints for perspective projection would be too unwieldy as the image appearance depends not only on the viewing angle but also the viewing position. We show that the resulting constraints, though more complex, are quite usable. Orthographic projection may be a good enough approximation when the viewing angle is small but perspective analysis may be necessary in other cases. However, we find that the perspective constraints actually carry *more* information and can provide more constrained or even unique interpretations.

Some researchers have investigated perspective projection before. Draper [Dra81] gave a constraint that derives from boundary between two faces. Sugihara [Sug86] gives a linear programming method to determine the realizability of a line drawing under orthographic projection. His formulation can also be carried out under perspective projection. However, this method leaves many degrees of freedom undetermined for surface orientations and does not provide

*This research was supported by the Defense Advanced Research Projects Agency under contract number F 33615-87-C-1436 monitored by the Air Force Wright Aeronautical Laboratories, Darpa Order No. 3119.

a way of incorporating other geometric constraints like symmetries. Sugihara does show how to use additional constraints such as shape from shading in an optimization scheme.

In this paper, we provide a set of techniques for perspective projection that parallel many of the traditional techniques for orthographic projection and hence can be applied where the latter techniques apply. Some of the constraints we describe have been previously presented by Shafer, Kanade and Kender [Sha83]; we will make specific references to their work in the appropriate places.

In section 2, we define some terms for perspective projection that are used to develop the constraints for perspective projection. Some of these constraints are generalizations of the constraints for orthographic projection, however, some are new and apply to perspective projection only. One of our contributions is the definition of a new kind of symmetry in the image that we call *convergent symmetry* that can provide unique orientations for such figures directly without using any other constraints! In section 4, we show how to apply the mathematical constraints we have derived for an example. In section 5, we give an analysis for curved surfaces.

2 PERSPECTIVE PROJECTION

Perspective projection is the exact projection model for the pinhole camera and a very good approximation for the lens systems used on cameras for objects in focus. In perspective projection there is a focal point, let it be the origin of the coordinate system. Any point, (x, y, z), in 3-D forms a ray passing through the point and the focal point (the origin). We can represent this ray as $(u, v, 1)$ where $u = x/z$ and $v = y/z$. The intersection of this ray with the $z = 1$ plane forms the image of this point, then the intersection has coordinates (u, v) on that plane. Note that any point (u, v) on the $u - v$ plane is also a point or a vector $(u, v, 1)$ in the $x - y - z$ coordinate system. This duality of the points will be used throughout the paper. Hereon the $z = 1$ plane is called the image plane, the projective plane or the $u - v$ plane.

Consider a line, $L = Rt + P$, in 3-D, parameterized in t, where $R = (r_x, r_y, r_z)$ is the orientation of the line and $P = (p_x, p_y, p_z)$ is any point on the line. From just the image of L on the projective plane we can not recover the parameters R or P. But we can extract some other useful parameters. The image, $L_i = R_i t + P_i$, of the line L, can be considered as a line in 3-D which lies on the $u - v$ plane. Say L_i has equation $au + bv + c = 0$ on the $u - v$ plane, then

$$R_i = (-b, a, 0) \quad P_i = (p_u, p_v, 1) \tag{1}$$

where P_i is any point on the line L_i and R_i is the 3-D orientation of the line L_i. Note that R_i and P_i are the only observables of the line L from the image of it, and they are not unique but scalable.

We define another useful observable. If the line L does not pass through the origin[1] then we can define a plane by the line L and the origin. This plane has the property that the image, L_i, of the line L is the intersection of this plane and the image plane. This plane will be called the *image generating plane* or IGP (this plane is called the interpretation plane by Macworth [Mac73]). IGP of L can also be constructed by using the line L_i and the origin. The normal of IGP (hereon it is called NIGP of the line L) A is given by:

$$\begin{aligned} A &= R \times P \equiv R_i \times P_i \\ &\equiv (a, b, -p_u a - p_v b) \end{aligned} \tag{2}$$

The $\equiv$ symbol indicates the parallelity of the vectors which implies componentwise equality up to a common scale. For $U = (u_x, u_y, u_z)$ and $V = (v_x, v_y, v_z)$, if $U \equiv V$ then $(u_x, u_y, u_z) = (\lambda v_x, \lambda v_y, \lambda v_z)$. The NIGP of a line is called the vanishing gradient of a line in [Sha83]. Note that $(-p_u a - p_v b)$ is equal to c since $(p_u, p_v, 1)$ is on the line L_i. In fact c is proportional to the minimum distance of the line from the origin of the $u - v$ plane. Then A is equal to:

$$A = (a, b, c) \tag{3}$$

The NIGP, A, of the line L is a very simple quantity that we can obtain directly from the equation of L_i on the image plane.

[1] If the line passes through the origin then it projects as a point on the image plane, therefore this plane is defined for every line visible on the image plane.

3 CONSTRAINTS UNDER PERSPECTIVE PROJECTION

We now derive several constraints that follow from the properties of lines and surfaces under perspective projection.

3.1 Choosing a representation for Surface Orientation

Consider a plane in 3-D having equation:

$$ax + by + cz + d = 0 \tag{4}$$

The normal of this plane is $N = (a, b, c)$. However, as the normal of the plane has only two degrees of of freedom, we can normalize the normal vector as $N = (p, q, 1)$, where $p = a/c$ and $q = b/c$ (note that this excludes cases where $c = 0$). (p, q) can be viewed as a point in the *gradient space.* Gradient space has been useful for orthographic analysis since the degeneracies of gradient space (normals of the planes parallel to the z axis) also corresponds to the degeneracies of the orthographic projection (those planes project as lines). However, planes that are unrepresentable by the gradient space may be present in perspective images. Unfortunately, up to date we were unable to find a representation for the normal of a plane having only two components and the condition that the equation of the plane be linear in these components. We will derive our constraints first in abstract vector notation and then give two different representations. First representation is the regular gradient space; it has the advantage of simplicity but contains important singularities. The second one is just a regular vector (p, q, r) in 3-D with the constraint that:

$$p^2 + q^2 + r^2 = 1 \tag{5}$$

This can represent normal of any plane in 3-D, with the added complexity that equation 5, a quadratic equation, should be included among equations to be solved.

3.2 Shared Boundary Constraint

This constraint relates the orientation of two planes intersecting using the image of the line of intersection. Similar results has been derived previously in [Sha83]. Say two planes have normals N_1 and N_2, then the line, $L = Rt + P$, formed by intersection of these planes has orientation:

$$R = N_1 \times N_2 \tag{6}$$

If the image $L_i = R_i t + P_i$ has the equation $au + bv + c = 0$ on the $u - v$ plane then the NIGP of the line is $A = (a, b, c)$. Also note that $A \equiv R \times P$, that is $A \perp R$, therefore:

$$\begin{aligned} A \cdot R &= 0 \\ A \cdot N_1 \times N_2 &= 0 \end{aligned} \tag{7}$$

This is the shared boundary constraint in the form of a vector equation. Depending on the representation for N_1 and N_2 the final equation changes, but the vector equation remains the same. If the gradient space is used with $N_1 = (p_1, q_1, 1)$ and $N_2 = (p_2, q_2, 1)$ then the shared boundary constraint becomes:

$$a(q_2 - q_1) - b(p_2 - p_1) + c(p_2 q_1 - p_1 q_2) = 0 \tag{8}$$

Here a, b and c are known and unique quantities up to a scale factor. This equation defines a line in $p - q$ space when we fix one of the normals N_1 or N_2.

If the (p, q, r) representation is used, then $N_1 = (p_1, q_1, r_1)$, $N_2 = (p_2, q_2, r_2)$ and the constraint equation is:

$$a(q_1 r_2 - q_2 r_1) + b(r_1 p_2 - p_1 r_2) + c(p_1 q_2 - q_1 p_2) = 0 \tag{9}$$

This equation defines a plane in terms of (p_1, q_1, r_1) when (p_2, q_2, r_2) is fixed or vice versa. In this representation there are actually three constraint equations, one is the above and two others are obtained by substituting (p_1, q_1, r_1) and (p_2, q_2, r_2) in equation 5.

3.3 Parallelity Theorem

In orthographic projection, parallel lines in 3-d project into parallel lines in the image. This, is not the case, in general, under perspective projection. However, *if* we are given the information that two image lines are in fact parallel in 3-D, we can infer some important information about their orientations.

Theorem 1 *If two lines $L_1 = R_1t + P_1$ and $L_2 = R_2t + P_2$ are known to be parallel in 3-D (i.e. $R_1 \equiv R_2 \equiv R$). Then the orientation of the lines, R, is given by $R = A_1 \times A_2$ where A_1 and A_2 are the NIGPs of the lines L_1 and L_2.*

Proof This is not an entirely new result but has been used previously in [Bar83], [Sha83]. A_1 and A_2 are computable from the image of the lines L_1 and L_2 as given by equation 3. Also from equation 2, $A_1 = R \times P_1$ and $A_2 = R \times P_2$., then we get:

$$\begin{aligned} A_1 \times A_2 &= (R \times P_1) \times (R \times P_2) \\ &= (R \cdot (P_1 \times P_2))R - (P_1 \cdot (R \times R))P_2 \\ &= (R \cdot (P_1 \times P_2))R \\ &\equiv R \end{aligned} \tag{10}$$

Unless the lines L_1 and L_2 are parallel to the image plane, their images intersect and the intersection point, I, is given by $I \equiv A_1 \times A_2$. Note that this theorem does not have any analogy in orthographic projection.

3.4 Orthogonality Constraint

This constraint is derived from the *knowledge* that two lines in a plane are orthogonal in 3-D. In orthographic projection, this hint may come from the observation of a skew symmetry. For perspective projection, we will assume the orthogonality knowledge to be given for now, in the next sub-section, we show how it may be inferred from a new form of symmetry that we call the *convergent symmetry.* This constraint could also be applied to curved surfaces where we may have some means of detecting lines of minimum and maximum curvatures.

Consider a plane Π having normal N, and two orthogonal lines on the plane

$$L_1 = R_1t + P_1 \quad L_2 = R_2t + P_2 \tag{11}$$

These lines have the NIGPs $A_1 = (a_1, b_1, c_1)$ and $A_2 = (a_2, b_2, c_2)$. Since A_1 is the normal of the plane containing L_1 and the origin and also L_1 is on the plane Π then L_1 is the intersection of these two planes. Therefore

$$R_1 = A_1 \times N \quad R_2 = A_2 \times N \tag{12}$$

By the orthogonality constraint (i.e. $R_1 \perp R_2$) we get:

$$\begin{aligned} R_1 \cdot R_2 &= 0 \\ (A_1 \times N) \cdot (A_2 \times N) &= 0 \end{aligned} \tag{13}$$

This is the orthogonality constraint in the form of a vector equation. This constraint takes slightly different form depending on the representation used. If we use the gradient space, then $N = (p, q, 1)$ and the constraint equation is:

$$\begin{aligned} (a_1a_2 + c_1c_2)q^2 + (b_1b_2 + c_1c_2)p^2 - (a_1b_2 + a_2b_1)pq - \\ (b_1c_2 + b_2c_1)q - (a_1c_2 - a_2c_1)p + b_1b_2 + a_1 + a_2 &= 0 \end{aligned} \tag{14}$$

This is a quadratic equation in terms of the gradient (p, q) of the plane Π. For most choices of parameters, this will represent a hyperbola on the $p - q$ plane, as for orthographic projection, but not necessarily centered at the origin.

If we use the (p, q, r) representation then $N = (p, q, r)$ and the constraint equation is:

$$\begin{aligned} (b_1b_2 + a_1a_2)r^2 + (c_1c_2 + a_1a_2)q^2 + (c_1c_2 + b_1b_2)p^2 + \\ ((b_1c_2 - b_2c_1)q + (-a_1c_2 - a_2c_1)p)r + (-a_1b_2 - a_2b_1)pq &= 0 \end{aligned} \tag{15}$$

Note that p and q in this representation are not the same as for the gradient space. This is a quadratic surface in $p - q - r$ space. As in the case of shared boundary constraint, this equation should be used in conjunction with the constraint equation 5 which is a sphere. With these constraints only one degree of freedom is left for the orientation of the plane Π.

As in the case of orthographic projection, the orthogonality constraint by itself does not give unique orientations. As before, some *ad hoc* choices could be made, or this constraint needs to be used in conjunction with others.

3.5 Convergent Symmetry

In this section an object refers to a planar surface in 3-D bounded with a piecewise linear boundary, and a figure refers to the projection of the boundary. An object is called *symmetric* in 3-D if there are lines on the object joining the points of the boundary, called lines of symmetry, such that the locus of the mid points of these lines forms another line, called axis of symmetry, and that the axis of symmetry is orthogonal to the lines of symmetry. An arrow like object and its symmetry axis with the lines of symmetry are shown in figure 1(a). If we project a symmetric object using orthographic projection we get a figure having skew symmetry as proposed by Kanade [Kan81]. If we use perspective projection then we get a figure having a new symmetry called *convergent symmetry.*

Definition: A figure is said to be *convergent symmetric* if there exist point to point correspondences between all points of the figure such that:

(a) All lines joining points of correspondence, called lines of symmetry, intersect in a common point on the image plane.

(b) The projection of the mid-points of the 3-D lines of symmetry lie along a straight line on the image plane.

Under perspective projection, projections of parallel lines meet at a point on the image plane. Therefore the projections of the lines of symmetry should meet at a point when extended on the image plane, that is they should be convergent. The axis of symmetry is, however, no longer defined by the locus of the mid points of the lines of symmetry in the image plane. Instead, we require that the midpoints of the lines of symmetry, in 3-D be along a straight line. We show how this 3-D computation can be performed in the following. Figure 1 (b) shows an example of an arrow like object under perspective projection with its axis and lines of symmetry. First, we give a formal definition for convergent symmetry and then a procedure for checking it.

The corresponding points would be easier to determine in a figure with several corners, as each corner must correspond to another corner. However, the above definition is general and applies to any figure (including curved figures). In general, of course, we can first choose the point of convergence, and then define lines of symmetry from it. The following procedure is to check that the second part of the definition is also satisfied.

For every line of symmetry we can find the projection of its 3-D mid point. Consider figure 1 (b). Let $L_l = R_l t + P_l$ be one of the lines of symmetry, and let E and F be the two corresponding points on this line with image coordinates of (u_e, v_e) and (u_f, v_f) respectively. Let (u_c, v_c) be the point of convergence for the lines of symmetry. Then $R_l = (u_c, v_c, 1)$ from the parallelity theorem, and $P_l = (u_e, v_e, 1)$ as L_l passes through E. With these values for R_l and P_l the $u - v$ coordinates of the image of a point on the line L_l is given by:

$$\left(\frac{u_e + tu_c}{1+t}, \frac{v_e + tv_c}{1+t}\right) \tag{16}$$

The coordinates of the image of the 3-D mid point of the line L_l between the points E and F is :

$$\left(\frac{(2u_e - u_c)u_f - u_c u_e}{u_f + u_e - 2u_c}, \frac{(2v_e - v_c)v_f - v_c v_e}{v_f + v_e - 2v_c}\right) \tag{17}$$

This gives us a procedure for finding the projection of the 3-D mid-point of any given line of symmetry. To check whether a given figure is convergent symmetric, we simply need to find the projections of mid-points of all lines of symmetry and check that they lie on a straight line, say L_s (L_s is the projection of the 3-D axis of symmetry). The NIGP value for L_s, $A = (a, b, c)$, can be obtained by

$$A = T \times P \tag{18}$$

where T and P are midpoints of any two distinct lines of symmetry (the mid-points are given by equation 17).

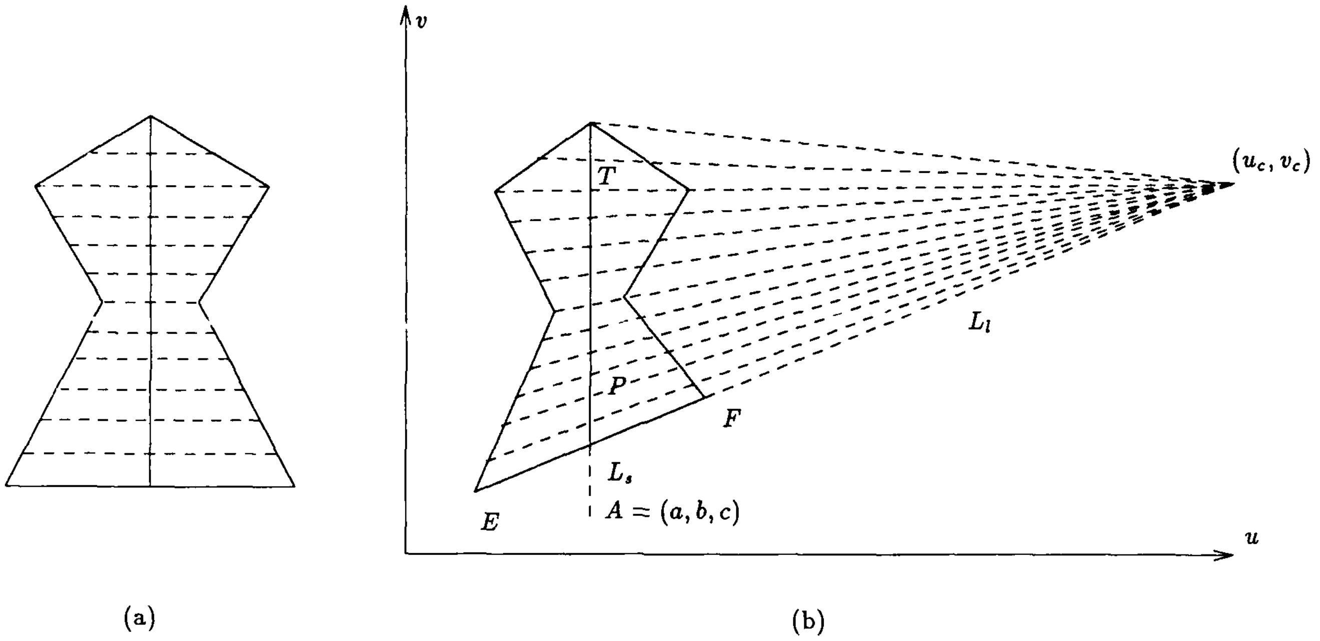


Figure 1: (a) An arrow like planar object with its axis of symmetry, solid vertical line, and lines of symmetry, dashed horizontal lines. (b) Projection of the arrow like object and its convergent symmetry lines; dashed lines are the lines of symmetry meeting at the point (u_c, v_c), L_l is one of the lines of symmetry meeting the boundary at points E and F. The vertical solid line is the axis of symmetry, L_s, having NIGP of $A = (a, b, c)$.

Computing Orientation Using Convergent Symmetry

Now we will apply the constraint that the axis of symmetry is orthogonal to the lines of symmetry in 3-D. First, we state a theorem related to this.

Theorem 2 *If a convergent symmetric figure is assumed to be a perspective projection of an orthogonal symmetric planar object, then the orientation of the planar object can be determined uniquely (unless the convergent symmetry is actually a skew symmetry with point of convergence at infinity, the axis of symmetry goes through the origin of the image plane and the lines of symmetry are orthogonal to the axis of symmetry on the image plane).*

We will give a constructive proof of this theorem in the following. Note that the theorem asserts that the constraints provided by convergent symmetry are much stronger than those provided by skew symmetry. The process is similar to that of skew symmetry analysis, but unlike in the case of orthographic projection, in perspective projection the axis of symmetry intersects every line of symmetry at a different angle than the others on the image plane. This results in a different constraint equation at every point on the axis of symmetry. Every equation gives a different constraints hyperbola on the $p - q$ plane (if $p - q$ space is used). But all of these hyperbolas goes through one point on the $p - q$ space and this point is the only solution to all of these constraints equation. Therefore, we get a unique answer for the surface normal by using convergent symmetry, except for some special cases noted in the theorem. In gradient space (p, q) representation, we can find a close form solution. And the degeneracies of $p - q$ space can be compensated as will be clear later. Consider the object in figure 1. Axis of symmetry has the NIGP of $A = (a, b, c)$ and assume that A is normalized (i.e. $|A| = 1$). There are infinitely many lines of symmetry all of which pass through the point (u_c, v_c) on the image plane. Say the intersection of these lines with the v axis has the coordinate $(0, k)$ where k is a parameter having a range that covers the figure. Then these lines have the NIGP:

$$\begin{aligned} A_l &= (0, k, 1) \times (u_c, v_c, 1) \\ &= (k - v_c, u_c, -u_c k) \end{aligned} \tag{19}$$

Say the normal of the plane containing the object is $N = (p, q, 1)$, then from the orthogonality constraint we have:

$$(A \times N) \cdot (A_l \times N) = 0 \tag{20}$$

$$\begin{aligned} & k((-cq^2 + bq - cp^2 + ap)u_c + aq^2 - bpq - cp + a)+ \\ & (-aq^2 + bpq + cp - a)v_c + ((-ap - c)q + bp^2 + b)u_c \quad = \quad 0 \end{aligned} \tag{21}$$

This constraint should be satisfied independent of the value of k, then we get two constraints of the form:

$$\begin{aligned} (-cq^2 + bq - cp^2 + ap)u_c + aq^2 - bpq - cp + a &= 0 \\ (-aq^2 + bpq + cp - a)v_c + ((-ap - c)q + bp^2 + b)u_c &= 0 \end{aligned} \tag{22}$$

There is only one real solution to these equations given by:

$$\begin{aligned} p &= \frac{(ab^2 + a^3)v_c^2 + (-b^3 - a^2b)u_c v_c + (-b^2 - a^2)cu_c - ac^2 + a}{(b^2 + a^2)cv_c^2 + (bc^2 - b)v_c + (b^2 + a^2)cu_c^2 + (ac^2 - a)u_c} \\ q &= -\frac{((ab^2 + a^3)u_c + (b^2 + a^2)c)v_c + (-b^3 - a^2b)u_c^2 + bc^2 - b}{(b^2 + a^2)cv_c^2 + (bc^2 - b)v_c + (b^2 + a^2)cu_c^2 + (ac^2 - a)u_c} \end{aligned} \tag{23}$$

This gives us the normal, $N = (p, q, 1)$, of the plane containing the object in terms of the observables, $A = (a, b, c)$, and the intersection point, (u_c, v_c), of lines of symmetry on the image plane. As mentioned before in the gradient space representation normal of the planes that are parallel to z axis are not representable, that is because those planes have the third component of the normal vectors equal to zero, and equivalent of a vector, $V = (f, g, l)$, under this representation is obtained by dividing the vector by the third component of the vector, $(f/l, g/l, 1)$. However, the expressions for p and q in equation 23 have the property that the denominator for p and q are the same then by multiplying the N vector with this denominator we get another vector, N', having the same orientation as N but have no singularity as for representing planes parallel to z axis. Then the vector N' is :

$$\begin{aligned} N' \quad = \quad & ((ab^2 + a^3)v_c^2 + (-b^3 - a^2b)u_c v_c + (-b^2 - a^2)cu_c - ac^2 + a, \\ & -((ab^2 + a^3)u_c + (b^2 + a^2)c)v_c + (-b^3 - a^2b)u_c^2 + bc^2 - b, \\ & (b^2 + a^2)cv_c^2 + (bc^2 - b)v_c + (b^2 + a^2)cu_c^2 + (ac^2 - a)u_c) \end{aligned} \tag{24}$$

Unlike the skew symmetry under orthographic projection, for a convergent symmetric figure in perspective projection we can compute the orientation of the planar surface completely. That is, we even do not have the neckers reversal, this is also in agreement with human perception. For example the cube in figure 2 can be reversed if one tries but the reversed figure does not look symmetric at all. Therefore if we want to bias towards symmetric objects then there is only one answer for a convergent symmetry figure. This is another instance in which the perspective projection can be used to give more information than the orthographic projection.

All of the above derivations assume that the intersection point (u_c, v_c) is not at infinity. In the latter case, we can obtain the solution using the limits of the solutions. Let us say the slope of the lines of symmetry is m, then we can obtain the solution by replacing v_c by mu_c in equation 23 and taking the limit as u_c goes to infinity. Then the solution in 3-component vector form is:

$$N' = (am^2 - bm, -am + b, cm^2 + c) \tag{25}$$

In the theorem of convergent symmetry, we mentioned that we get a unique orientation from convergent symmetry except under some special cases. In fact, if lines of symmetry are parallel to each other on the image plane, and image of the axis of symmetry is passing through the origin of the image plane (i.e. $c = 0$), and on the image plane the axis of symmetry is orthogonal to the lines of symmetry (i.e. $b/a = m$), then N' becomes a zero vector. However this requires a very specific viewing angle and thus can be ignored under normal viewing conditions. This is the case that convergent symmetry acts like the skew symmetry of orthographic projection, that is, now it is a constraint leaving one degree of freedom, which is basically the orthogonality constraint given in equation 13.

4 USAGE OF THE CONSTRAINTS FOR POLYHEDRAL OBJECTS

In the previous section we have derived four constraints under perspective projection (however, not all four are independent as parallelity and orthogonality constraint are used in the convergent symmetry). For a given figure, we need to determine which constraints are applicable. Note that the shared boundary constraints make no regularity assumptions about the figure and must always apply (ignoring any "errors" in the line drawing).

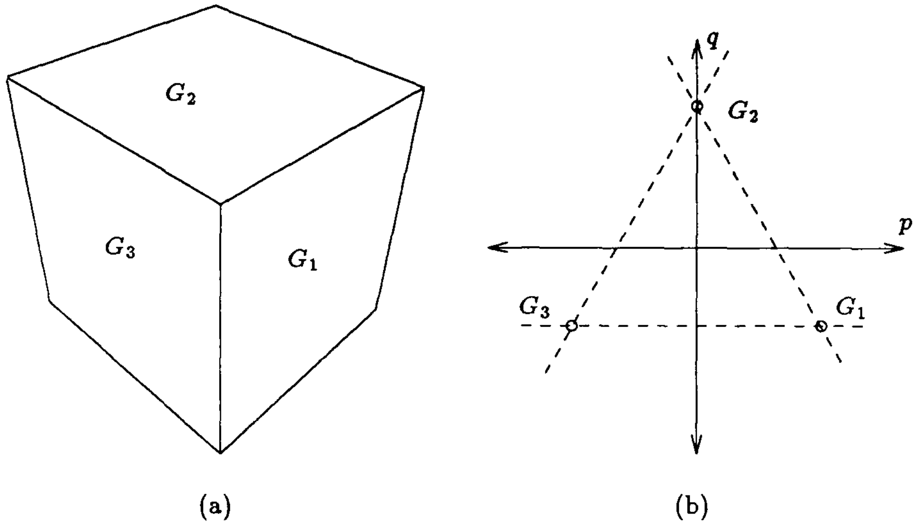


Figure 2: A cube under perspective projection (a), and computed orientations for the faces shown as points on the $p - q$ space with the shared boundary constraints overlaied, dashed lines, (b).

Other constraints, however, require observation of some regularity in the image and assumption that the 3-D object obeys corresponding regularity also. Of these regularities, symmetric convergence is quite stringent, *i.e.* it is unlikely to be caused by accident, though we still can not guarantee that the 3-D object is orthogonally symmetric. Unfortunately, symmetric convergence is strong only when at least three lines converge on the image plane, as two lines always converge (unless they are parallel to eachother on the image plane). Thus, a planar object .iaving four sides to a face can always be construed to be symmetric convergent. Observations about parallelism and orthogonality may also not be apparent in the image. In orthographic projection, parallel lines remain parallel; in perspective projection they do not. On the other hand, however, in perspective projection we have much tighter constraints. Thus, one way to solve the interpretation problem is to *make* regularity assumptions and *verify* by using the constraints. We illustrate this by an example.

Figure 2(a) shows the image of a cube under perspective projection (the reader will get a better perception of the figure if the picture is held very close to the eye). Applying the shared boundary constraint (in the gradient space for the sake of illustration here) gives us a triangle, say $G_1G_2G_3$ in figure 2(b) which specifies the orientations of the three faces. Note that in perspective projection, the shape of the triangle may depend both on its position and its size (both of which need to be determined). Additional constraints can come from the symmetry of the faces. As described earlier, any quadrilateral can be viewed as being convergent symmetric. Assuming that the three faces are projections of orthogonally symmetric shapes (*i.e.* rectangles), we can get unique orientations for the three faces. In this example, these values happen to be consistent with the shared boundary constraints (and with the known correct answers from which the example was constructed). Alternately, we could have used the parallelity constraints between opposite sides of the faces. For this example, this regularity is suggested by the observation that groups of *three* lines (corresponding to parallel lines on two faces) do intersect in common points. Using this constraint, the answers turn out to be the same as before and hence consistent.

Of course, in general, we can not expect all constraints to be satisfied simultaneously for all figures. If the image had been derived from a rhombus, instead of a cube, the convergent symmetric results would not agree with the shared boundary constraints. Now, we can make several choices. We can either make all faces equally non-symmetric (by some measure), or still achieve consistency by making two faces (any two) symmetric and the third non-symmetric. In general, it should be possible to define a penalty function and find "optimal" solutions. However, we have not investigated such approaches. Our feeling is that the only time we can get strong interpretations is when some of the evidence is overwhelmingly strong and that this is the evidence we would use at the exclusion of the other constraints (those that require some assumptions).

5 EXTENSIONS TO CURVED SURFACES

In a recent paper we [UN88] described methods for recovery of surface orientation of curved surfaces from contours under orthographic projection. The analysis was based on observation of a form of symmetry that we called *parallel*

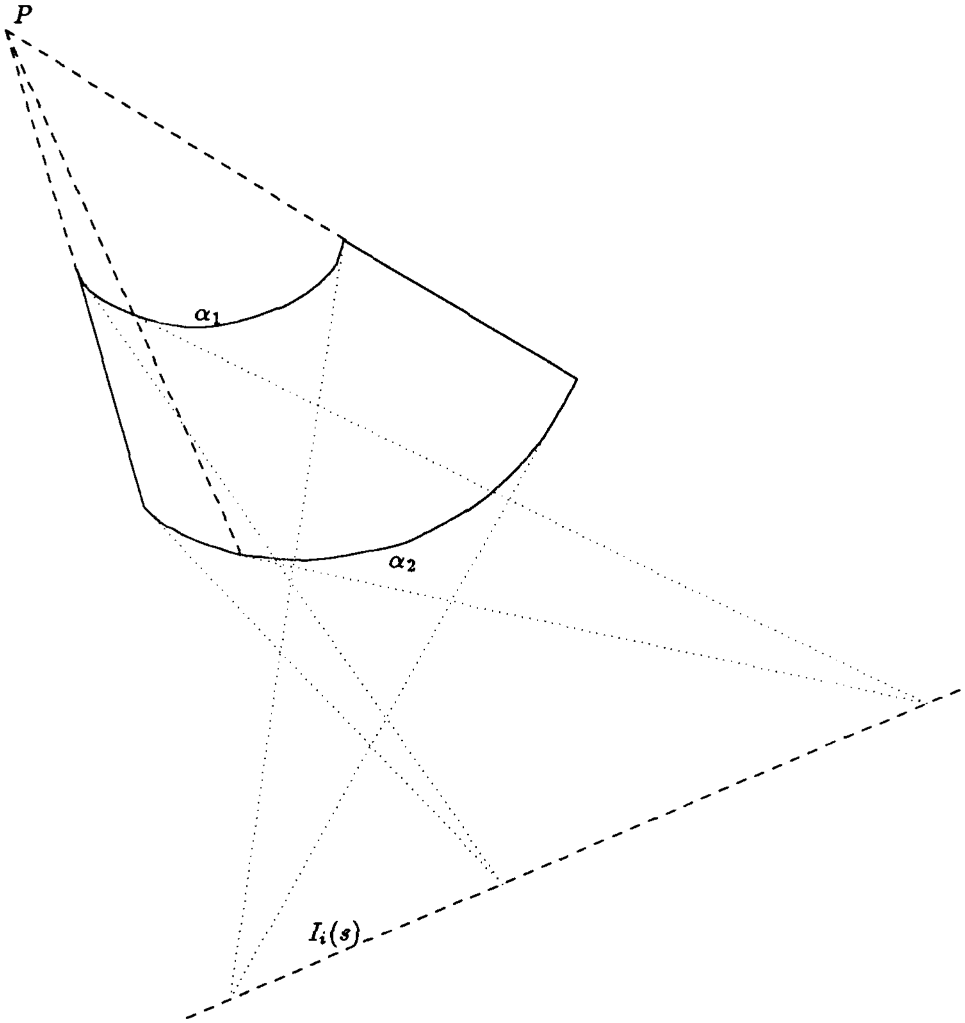


Figure 3: Contours of a conic surface under perspective projection, with the point of convergence for the rulings P, and the line $I_i(s)$.

symmetry. Two planar curves are defined to be parallel symmetric if there exist a one to one correspondence between the points on the curves such that the tangents to the curves at corresponding points are parallel. The importance of the parallel symmetry is that, if we cut a zero gaussian curvature surface with two parallel planes. Then it can be shown that we get two parallel symmetric curves from the intersection of the planes with the surface such that corresponding points of these curves are joined by the rulings of the zero gaussian curvature surface.

Two curves that are parallel symmetric in 3-D also project into parallel symmetric curves in the image under orthographic projection. However, this is not the case under perspective projection. In the following, we give conditions that curves in a perspective image must satisfy if they are projections of parallel symmetric 3-D curves. Then we give a method for computing parallel symmetry for certain class of images (those that are projections of conic surfaces) and show how this can be used for surface reconstruction.

5.1 Parallel Symmetry

Say there are two curves α_1 and α_2 on the image plane generated by the two planar 3-D curves β_1 and β_2 in planes parallel to plane, call it Π, which passes through the origin. We have the relation:

$$\begin{aligned} \alpha_1(s) &= \frac{\beta_1(s)}{\beta_{1z}(s)} \\ \alpha_2(s) &= \frac{\beta_2(s)}{\beta_{2z}(s)} \end{aligned} \tag{26}$$

were β_{i_z} is the third coordinate of the curve β_i. The curves β_1 and β_2 are parallel symmetric iff we can form a monotonic correspondence function $f(s)$ such that:

$$\beta_1'(s) = \beta_2'(f(s)) \tag{27}$$

where $\beta_i(s)$ is the tangent vector of the curve $\beta_i(s)$. For each point of the curve α_i the NIGP of the tangent line (i.e. the line passing from the point $\alpha_i(s)$ in the direction $\alpha_i'(s)$) of the curve is $A_i = \alpha_i(s) \times \alpha_i'(s)$. Since the curves β_1 and β_2 have parallel tangents at the corresponding points. From the prallelity theorem the vector function $I(s) = A_1(s) \times A_2(f(s))$ gives the orientation of the tangents of the curves β_1 and β_2. That is

$$\beta_1'(s) = \beta_2'(f(s)) \equiv I(s) = A_1(s) \times A_2(f(s)) = (\alpha_1(s) \times \alpha_1'(s)) \times (\alpha_2(f(s)) \times \alpha_2'(f(s))) \tag{28}$$

Since the curves β_1 and β_2 are planar curves resting on planes parallel to Π, their tangents should be on the plane Π. Therefore every orientation given by the function $I(s)$ (i.e. the vector from the origin to the points of $I(s)$) should be on the plane Π. The image of $I(s)$, $I_i(s)$, is the curve on the image plane that can be obtained by projecting the points of $I(s)$ on to the image plane. Since $I(s)$ is on the plane Π which passes through the origin its image has to be a line. $I_i(s)$ being a line is the necessary condition that two curves α_1 and α_2 are projections of the parallel symmetric curves β_1 and β_2. Also the orientation of the plane Π is just the NIGP of the line $I_i(s)$ since Π is the IGP of this line. Also the curve $I_i(s)$ is the locus of intersection points of the tangent lines of the curves $\alpha_1(s)$ and $\alpha_2(f(s))$ see figure 3.

5.2 Analysis of a Conic Surface

We now concentrate on conic surfaces (or linear straight homogeneous generalized cones in generalized cones terminology) cut by two parallel planes (say parallel to plane Π) to form curves, say $\beta_1(s)$ and $\beta_2(s)$. Let α_1 and α_2 be the projections of β_1 and β_2. In this case, the curves $\beta_1(s)$ and $\beta_2(s)$ are parallel symmetric; let the correspondence function be $f(s)$ as before. The lines joining the corresponding points on the curves β_1 and β_2 are the *rulings* of the surface. For a conic surface, these rulings intersect in a single point in 3-D. For a cylindrical surface, these rulings are parallel to each other. In either case, the projections of the rulings intersect at a point, say P, in the image plane.

Point P can be found by the intersection of the lines joining the end-points of α_1 and α_2. Now draw lines from P such that they intersect curves α_1 and α_2; the intersection points are the corresponding points on the two curves. With this correspondence we can construct the curve $I(s)$ and check if $I_i(s)$ is straight as in figure 3. If it is, we can interpret the figure as a conic surface. (Note: point P can also be found by a search process if the end-points of the curves are not reliable.)

Now we have the plane Π containing $I(s)$, we can reconstruct β_1 and β_2 by backprojecting α_1 and α_2 onto planes parallel to Π (up to a scale). However, this is not sufficient to reconstruct the conic surface; the distance between the planes containing the two curves still remains as a one degree of freedom.

This degree of freedom can be fixed if we interprete the surface as being cylindrical (under perspective, a conic surface can always be interpreted as being cylindrical). Given the orientation N of the plane Π containing 3-D curves $\beta_1(s)$ and $\beta_2(s)$, the 3-D tangent $\beta_i'(s)$ is given by:

$$\beta_i'(s) = A_i(s) \times N \tag{29}$$

where $A_i(s)$ is the NIGP of the line passing through $\alpha_i(s)$ in the direction $\alpha_i'(s)$ $A_i(s) = \alpha_i(s) \times \alpha_i'(s)$, therefore

$$\beta_i'(s) = (\alpha_i(s) \times \alpha_i'(s)) \times N \tag{30}$$

Now we have the orientation of $\beta_i(s)$ at any point. The orientation of the surface is given by :

$$\beta_1'(s) \times R(s) \tag{31}$$

where $R(s)$ is the 3-D orientation of the rulings. Since the surface is cylindrical, $R(s)$ is constant (i.e. $R(s) = R$) and is equal to the intersection point P of the rulings on the image plane (c.f. parallelity theorem). That is $R \equiv P$ and the orientation of the surface at any point is given by:

$$((\alpha_1(s) \times \alpha_1'(s)) \times N) \times P \tag{32}$$

We conjecture that if the resulting surface corresponds to a right cylindrical surface[2] humans will accept this interpretation as being the most preferred. If the figure is to be interpreted as a non-cylindrical conic surface, further assumptions need to be made. One alternative is to assume that the surface belongs to a right, generalized cone. We have not studied the human preferences in such cases, and such experiments are in fact difficult to perform.

6 CONCLUSION

We have derived some constraints on the interpretations of line drawings under perspective projection. Some of the constraints are analogous to the constraints used in orthographic analysis. However, in perspective analysis, we typically need to use one more variable in representing orientations; this makes some of the equations more complex and non-linear. The observation of the regularities may also be not as clear with perspective as it is with orthography. However, when such regularities can be inferred from some other, perhaps external, context, our constraints can be used directly.

Our major observation, however, is that when regularity is discovered in perspective, it provides much stronger constraints than under orthographic projection. We demonstrated this for the case of convergent symmetric figures. For the case of curved surfaces, too, perspective projection provides considerable amount of information. If the surface is cylindrical, (or can be interpreted as being cylindrical), than we can reconstruct the surface just from its contours. For conic surfaces one degree of freedom is left if do r.ot make additional assumptions. We hope that these observations will lead to increased use, and exploitation, of perspective projection rather than regarding perspective as a complicating agent that can be ignored under "norma!" viewing conditions.

References

[Asa87] M. Asada. Cylindrical shape from contour and shading without knowledge of lighting conditions or surface albedo. In *Proceedings of the First International Conference on Computer Vision*, pages 412–416, 1987. London.

[Bar83] S. T. Barnard. Interpretting perspective images. *Artificial Intelligence*, 21:435–462, 1983.

[BT81] H.G. Barrow and J.M. Tenenbaum. Interpretting line drawings as three dimensional surfaces. *Artificial Intelligence*, 17:75–116, 1981.

[BY84] M. Brady and A. Yuille. *An extremum principle for shape from contour. IEEE Transactions on Pattern Analysis and Machine Intelligence*, 6:288–301, 1984.

[Clo71] M.B. Clowes. On seeing things. *Artificial Intelligence*, 2(1):79–116, 1971.

[Dra81] W. S. Draper. The use of gradient and dual space in line drawing interpretation. *Artificial Intelligence*, 17:461–508, 1981.

[Huf71] D.A. Huffman. Impossible objects as nonsense sentences. *Machine Intelligence*, 6:295–323, 1971.

[Kan81] T. Kanade. Recovery of the three-dimensional shape of an object from a single view. *Artificial Intelligence*, 17:409–460, 1981.

[Mac73] A.K. Mackworth. Interpretting pictures of polyhedral scenes. *Artificial Intelligence*, 4:121–137, 1973.

[Sha83] S.A. Shafer. Shadow geometry and occluding contours of generalized cylinders. Technical Report Report CS-83-151, Carnegie-Mellon University, May 1983.

[Ste81] K. A. Stevens The visual interpretations of surface contours. *Artificial Intelligence*, 17:47–73, 1981.

[Sug86] K Sugihara. *Machine Interpretation of Line Drawings*. MIT Press, 1986.

[TX87] S. Tsuji and G. Xu. Inferring surfaces from boundaries. In *Proceedings of the 1st ICCV*, pages 716–720, 1987. London.

[UN88] F. Ulupınar and R. Nevatia. Using symmetrries for analysis of shape from contour. In *Proceedings of the 2nd ICCV*, 1988. Florida.

[Wei88] I. Weiss. 3-d shape representation by contours. *Computer Vision, Graphics and Image Processing*, 41:80–100, 1988.

[2] A right cylindrical surface is the one where planes cutting the surface to generate the curves β_1 and β_2 are orthogonal to the orientation of the rulings. This translates to the condition that $P \equiv N$, where N is the normal of the plane Π and P is the point on the image plane at which rulings intersect

Viewpoint Planning: the Visibility Constraint

Kostantinos Tarabanis[1]
Computer Science Department
Columbia University
New York, NY 10027

Roger Y. Tsai
Manufacturing Research
IBM T. J. Watson Research Center
Yorktown Heights, NY 10598

ABSTRACT

We present a new approach to the problem of sensor placement planning satisfying the visibility constraint, namely where the camera should be placed so that a target can be viewed without being occluded. The approach uses a new way of decomposing the general visibility planning problem into convex subtasks, and a new way of doing these convex subtasks. The decomposition not only drastically reduces the number of convex subtasks needed, but it also provides a natural *pruning* mechanism for further reducing their number by making the viewing region more *conservative*, that is, the resultant viewing region is a filtered version of the true one, but is contained completely inside the true one. Extensive implementation results are presented. The validity of the method is tested by placing the viewpoint inside the experimentally constructed viewing region, and seeing whether the target is truly visible. The accuracy of the boundary of the constructed viewing region is tested by placing viewpoints at the critical locations where the target is barely seen and observing whether the side wall of the viewing region is tangent to the line of sight. The results confirm that the method is accurate and valid.

Introduction

One of the major factors contributing to the development cost and time for machine vision applications is the determination of the placement of the camera and the associated optical setup. Being able to automatically determine the sensor placement is very important for reducing the development cycle and cost in today's manufacturing environment. Furthermore, with the increasing emphasis on process control and the associated measurements, it has become more and more important to have a machine vision system that automatically adapts itself to the changing process requirements, which are updated frequently during process optimization. The ability to automatically determine sensor placement given any process requirement is therefore desirable.

In this paper, we describe a new approach for planning sensor placement that avoids optical occlusion given any specified polyhedral target and occluding polyhedral solid opaque object. The algorithm applies to the general situation where the occluding object need not be convex, but the target is assumed to be convex. In Ref. 3, we describe the algorithm for the general case where both the occluding object and the target are non-convex.

Outline of The New Approach

The algorithm described in this section is not intended to be complete. The details are given in the sections "Algorithms for the Decompositions" and "Convex Visibility Subproblems" .

Existing approaches to this problem can be found in Ref. 1 and Ref. 2.

Motivation

Since the general non-convex viewpoint planning problem is very difficult, decomposing it into convex subtasks is a useful thing to do. However, using existing schemes, one formidable obstacle is the immense number of convex partitions of the target and occluding object that may exist in a real situation. So, one of the original motivations for the new approach described in this paper is to find a new way of decomposing the problem into convex subtasks that is generally more feasible. We found a decomposition called "Material-Hole Decomposition" that is ideal for this purpose, to

[1] this author was supported in part by DARPA contract N00039-84-C-0165 and in part by Manufacturing Research, IBM T.J. Watson Research Center.

be described in the next section. It yields a vastly smaller number of convex subtasks. As for the convex viewpoint planning subtask itself, we also found a new scheme called "the Rolling Method" that is far more efficient and simpler than the general half space intersection approach used in Ref. 1.

Material-Hole Decomposition (Loop Decomposition)

Figure 1 shows an occluding polygon with a hole in it. A straightforward decomposition (e.g. triangulation) will result in a large number of convex subtasks. However, we can actually do it with only two convex subtasks. This follows from the observation that the occluded region for viewing a target T with an occluding polygon B containing a hole H in it is equal to the occluded region caused by B (shown in Figure 2) LESS the region in space where the camera can view the target T through the hole H (shown in Figure 3). The first subtask is to determine the occluded region caused by the *material* polygon B. The second subtask is to determine the viewing region through the hole H. The resultant occluded region is equal to the difference of the above two regions, shown in Figure 4. In general, the following formula holds:

$$Occluded\ Region_{resultant} = Occluded\ Region_{material} - ViewingRegion_{hole\ within\ material} \qquad (1)$$

However, we actually need two levels of decomposition to accomplish the task in general. Observe that if the occluding polygon B or the hole H within B or the target T are not convex, then the two subtasks are still not convex, and therefore, we need another level of decomposition, as shown in the next section.

Convex Material-Gulf Decomposition (Convex Decomposition)

Consider the situation in Figure 1 discussed earlier. Suppose that the occluding polygon B is not convex and is shaped like that in Figure 5. Then it is necessary to decompose B into its convex hull B_{hull} and a gulf B_{gulf} such that

$$Polygon = Convex\ Hull - Gulfs \qquad or \qquad B = B_{hull} - B_{gulf} \qquad (2)$$

One might think that just as in the material-hole situation, the following should hold:

$$Occluded\ Region_{resultant} = Occluded\ Region_{convex\ hull} - Viewing\ Region_{gulf}$$

However, this is actually not quite right. Figure 6 shows the resultant occluded region. Clearly, the region viewed through the gulf is enclosed, which is not intuitively true. Actually, had the above been true, the resultant viewing region would have been the same as if the gulf in Figure 5 had its opening or mouth enclosed with an edge or line. Clearly this is not correct since there are valid viewpoints such that the target is fully visible but it is viewed only partially through the gulf and partially through the space outside the convex hull B_{hull}. To obtain the correct viewing region, the gulf should be enlarged to be $B_{gulf\_virtual}$ or B_{equiv_hole} as shown in Figure 7. Although B_{equiv_hole} is not convex, the viewing region can still be simply computed, as to be shown in "Gulf Viewing Region Computation". The resultant viewing region through the gulf is shown in Figure 8, and the resultant occluded region is shown in Figure 9.

Convex Visibility Planning

After the above two decompositions (loop and convex), what remains is the basic convex visibility planning task. There are two kinds of convex tasks, one is for occluded region computation, and the other for viewing region computation. We have found a new Rolling Method that computes the occluded region efficiently. A "hinge-and-swing" method is used to compute the viewing region. In either case, no general half space intersection is needed.

Algorithms for the Decompositions

Loop Decomposition

Polygons in general may contain loops (cycles of edges) inside their outermost boundary. These loops can be nested, as shown in the polygon of Figure 10 . The loop decomposition partitions the original polygon into the loops that it contains and builds a tree that represents their nesting explicitly.

Consider the polygon in Figure 10 . Loops p1 through p8 are nested as shown in the loop tree of Figure 10 . An edge of this tree indicates that the child loop, p3 for instance, is contained in its parent loop, p1. As a result, the height of the loop tree represents the degree of nesting inside the polygon.

In addition, the interior of loops at odd levels of the tree, like p1,p5,p6 and p7, are material regions, while loops at even levels, such as p2,p3,p4 and p8, are holes. We name the former, material loops and the latter, hole loops.

Convex Decomposition

After the loop decomposition is done, another level of decomposition, the convex decomposition, is needed to divide the task into convex viewing subtasks.

The convex decomposition algorithm approximate, a simple polygon (a polygon with a single loop) by a sum of convex polygons. These convex polygons can be added and subtracted in an alternating sequence to construct the original polygon.

Consider the polygon of Figure 11 . As a first step, the convex hull C_1 , shown in Figure 12 approximates the initial polygon and is convex. However, this clearly overestimates the polygon by the area equal to the difference between its convex hull and the polygon itself. In general, this difference consists of concave polygons which in turn can be decomposed in a similar fashion. At the second stage then, the convex hull of each polygon in this difference is computed, resulting in the convex polygons C_2, C_3, and C_4, shown in Figure 13 . At this point, when the convex polygons of stage two are subtracted from the convex polygon of stage one, the initial polygon is underestimated. The difference is again computed and the algorithm proceeds similarly, as shown in Figure 14 and Figure 15 .

The result of this decomposition is a set of convex polygons that can be arranged in a tree, which we call the convex tree. The original concave polygon can be generated by subtracting the convex polygons corresponding to children nodes from the convex polygon of their parent node, in a bottom-up fashion. In this way, convex polygons at odd levels of the tree are added to the sum that generates the original concave polygon (material polygons), while convex polygons at even levels of the convex tree (gulf polygons) are subtracted from this sum.

Applying this convex decomposition to the polygon of Figure 11 generates the convex tree shown in the same figure. In general, the height of the convex tree can be considered to represent the degree of concavity of the polygon.

Convex Visibility Subproblems

The general visibility problem between a target and an occluding polygon can now be reduced, after the above decompositions, to visibility subproblems between convex polygons.

There are three types of convex visibility subproblems that result:

- Occluding region computation
- Hole viewing region computation
- Gulf viewing region computation

Once these individual subproblems are sol:ed, what remains is to combine in an appropriate manner the component viewing and occluding regions for a particular occluding polygon and target pair. It is intuitively true that at any stage in the computation, occluded regions are added, while viewing regions are subtracted, to compute the final occluded volume. However, it is important that these component regions be considered in the proper order to guarantee correctness of the end result. The visibility computations need to start from the lowest levels of both the loop and convex trees ascending in a breadth-first manner to the root. This method builds the result in a "smallest-first" approach that ensures that regions are first combined locally. The global algorithm is given in more detail in Ref. 3.

Occluded Region Computation

Consider the occluding polygon and target shown in Figure 1 . Any plane that partitions three-dimensional space into two half-spaces, a half-space containing the target and the other, which subsumes the occluding polygon, has the property that the target is visible from any viewpoint chosen in the first half-space. In the limit, this plane may share an edge with the target and a vertex with the occluding polygon or vice-versa, in which case the viewing region attains a maximum. The occluding region is therefore bounded by a family of such limiting separating planes that are defined by an edge and a vertex, one from each polygon (in cases where the target and occluding polygon are properly aligned, the limiting separating plane is defined by two edges, one from each polygon).

We determine these separating planes by using a "rolling" method. In this approach, a separating plane is "rolled" between the target and the occluding polygon remaining tangent to both continuously. The limiting positions of this plane define the bounding planes of the occluded region.

A first limiting separating plane is found in the following way: The plane of the target is rotated around one of its edges until a vertex on the occluding polygon is encountered. At this point, a limiting separating plane is found. This plane, shown in Figure 16 to pass through the points A, B and C, is then rotated around the line BC constructed between the limiting vertex C and any of the two edge vertices, B in this case. The direction of rotation is shown in Figure 16. If the other edge vertex, A, was chosen, the rotation direction would be reversed. During this rotation either vertex D1 of the occluding polygon or D2 of the target will be encountered first by this plane. The rotation axis together with this vertex define the second limiting separating plane shown in Figure 16 to pass through points B, C and D1. A similar rotation is then applied to this new plane to determine the next separating plane.

After the family of separating planes is found, they are then intersected sequentially, thus avoiding any general half-space intersection. These lines of intersection together with the occluding polygon itself define the occluded region.

At this point, it is clear why the convexity of the target and occluding polygons is an inherent requirement for the rolling procedure. Only then do the limiting separating planes truly partition the viewing space into a visible region on one side of this plane and an occluded region on the other side.

Hole Viewing Region Computation

Consider now the hole polygon and target shown in Figure 1 . If, for each edge of the hole polygon, a plane is constructed that contains this edge and partitions three-dimensional space into two half-spaces, one of which contains both the target and the hole, then the target is visible from any viewpoint chosen in the intersection of these half-spaces. In the limit, these planes may share a vertex with the target in which case the viewing region attains a maximum. The viewing region is therefore bounded by a family of such limiting planes that are defined by an edge of the hole polygon and the associated limiting vertex on the target.

We find these limiting planes in the following way:

The plane of the hole is rotated around each of the edges of the hole until the last vertex on the target polygon is encountered. At this point a limiting separating plane has been found. In this case, the limiting planes are such that the target and hole are now in the same half-space, as opposed to different half-spaces in the occluding case.

These limiting planes are then intersected sequentially, again avoiding any general half-space intersection. The lines of intersection together with the hole polygon itself define the viewing region.

Gulf Viewing Region Computation

Consider the gulf and target shown in Figure 5 . As explained in "Convex Material-Gulf Decomposition (Convex Decomposition)" , we reduce the problem of determining the viewing region associated with a gulf to that of computing the viewing region of an equivalent hole.

The equivalent hole is larger than the gulf, and is extended from the the gulf to the outside of the convex hull of the occluding polygon (see Figure 7). The algorithm for the construction of the equivalent hole can be found in Ref 3.

The equivalent hole shown in Figure 7 is clearly concave and is therefore decomposed, like any other hole loop, into its convex parts. In turn, the visibility regions of these convex parts are then combined to determine the viewing region of the equivalent hole, shown in Figure 8 .

The Advantages of the New Method

It has been described earlier that the global task as well as the occluding object are represented by a tree-within-a-tree structure. This tree provides a good mechanism for pruning for the sake of speed. Since the nodes close to the bottom of the tree (especially the convex tree) represent fine details both for the final viewing region and for the occluding object, pruning the tree by eliminating nodes close to the bottom becomes a natural 'filtering" process both for the task and for the final viewing region. This is useful since for computing the viewing region, it is not necessary to determine the boundary of the viewing region very precisely, although it should be *conservative* , in the sense that the viewing region can be slightly smaller and simpler than the true viewing region, but for all points inside the viewing region, the visibility constraint is satisfied. Therefore, the pruning must start from the level of the tree that represents holes (holes increase the viewing region while material does the opposite). For objects with many minute details, pruning becomes quite important to make the computation feasible.

Another advantage of the new method is its speed. There are two factors that contribute to the speed. The first comes from the fact that the new method produces much fewer convex subtasks than existing methods, as explained in "Outline of The New Approach". The second comes from the pruning mechanism explained above.

Test Results

We have implemented a working system for visibility planning. In this section, we seek to demonstrate that the results produced by the working system, which incorporates the new method described in this paper, are correct. We do this two ways. One is to show the 3-D view of the occluded region for some typical examples of occluding 3D polyhedra and targets, so that visually, these results seem plausible and reasonable. Then, we demonstrate the validity of the results by moving the viewpoint to some spot in the computed viewing region, and show the perspective view of the occluding object and the target. If the computation for the viewing region is correct, then the target should be visible. In the following, we first describe the environment and experimental setup. Then, we present the two ways of showing the correctness of the working system described above.

Environment

Our algorithm was implemented in AML/X, an object-oriented programming language intended for use in design and manufacturing applications. The programs are run in the TGMS (Tiered Geometric Modeling System) environment (Ref. 4.). TGMS provides an object-oriented programming interface to our in-house solid modeling system, GDP (Geometric Design Processor) (Ref. 5), as well as many geometry classes and methods.

In this framework, the occluding and target objects as well as the viewing and occluded regions, are represented as solids and any operations on them (e.g. convex hull, boolean set operations), are conveniently developed.

Test Results when the Occluding Object is Three-Dimensional

In Figure 17 an occluding polyhedron and a target are shown. The occluding polyhedron is first decomposed into faces. Each face is then treated as a separate occluding polygon.

Consider faces F_{top} and F_{bot} in Figure 17 . Their corresponding occluded regions are shown superimposed in Figure 18. After the union of these two regions is taken, then the viewing region associated with the hole is reduced to the region where the target can be viewed through both the top and bottom faces of the polyhedron. The union of this partial result with the occluded regions of the remaining faces of the polyhedron produces the final occluded volume of the polyhedron (see Figure 19).

Validation

As mentioned earlier, the purpose of validation is to place the viewpoint inside the experimentally constructed viewing region, and see whether the target is truly visible. We are going to use the three dimensional example of the previous paragraph and shown in Figure 17. The viewing area through the hole is considered and for this area, we are going to show two views. One view is the *comfortable view* where the target is viewed with some margin of clearance between the occluding object and the target. Another view is the *critical view* where the occluding object just barely clears the target. The purpose for choosing the critical view is for validating the preciseness of the boundary of the viewing region. If the boundary is precise, then the side wall of the viewing region should be tangent to the line of sight, making the side wall invisible. Such is indeed the case, as we shall see.

Figure 19 and Figure 20 show the *comfortable view* and the *critical view* respectively. It is seen that the target is clearly visible, and that the side wall of the viewing region for the critical viewing condition vanishes, confirming that the boundary of the viewing region is accurate.

Other test results and a more extensive validation of the computed occluded region can be found in Ref. 3.

Conclusion

A new method for viewpoint planning satisfying the visibility constraint as well as results of experimentation and validation have been presented. The method applies to the general case where the occluding object need not be convex. In this paper, the target is assumed to be convex. In Ref. 3, we describe the algorithm for the general case where the target is not convex. We will also consider extending the method to include curved surfaces. Currently, if the method described in this paper is to be applied as is, then curved surfaces must be approximated by polyhedra containing the curved surfaces completely so that the viewing region is *conservative* and *valid*.

References

1. "Automatic Sensor Placement from Vision Task Requirements", Cowan C., and Kovesi P., SRI report, Menlo Park, CA, June 1987.
2. "Model-Based Planning of Visual Sensors Using a Hand-Eye Action Simulator System: Heaven", Sakane S., Sato T., and Kakikura M., Electrotechnical Laboratory report, MITI, Japan, 1987.
3. "Occlusion Free Sensor Placement Planning", Tsai R. Y., and Tarabanis K., Proceedings of Third Annual Machine Vision Workshop, New Brunswick, NJ, April 3-4, 1989.
4. "TGMS: An Object-Oriented System for Programming Geometry", Dietrich W., Nackman L.R., Sundaresan C.J., Gracer F., IBM Research Report, IBM T.J. Watson Research Center, Yorktown Heights, NY, January 1988.

5. "A Geometric Modeling System for Automated Mechanical Assembly", Wesley M. A., Lozano-Perez T., Lieberman L. I., Lavin M. A., Grossman D. D., IBM Journal of Research and Development, January 1980.

6. "Computational Geometry", Preparata, F., and Shamos M., Springer Verlag, 1985.

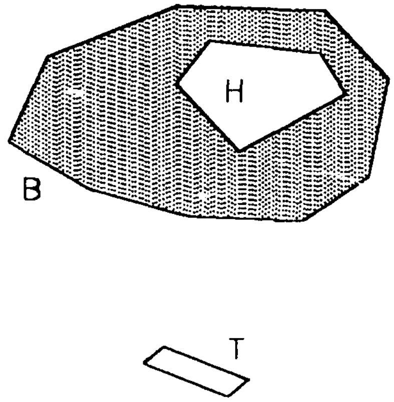


Figure 1. The occluding object is a polygon with a hole in it.

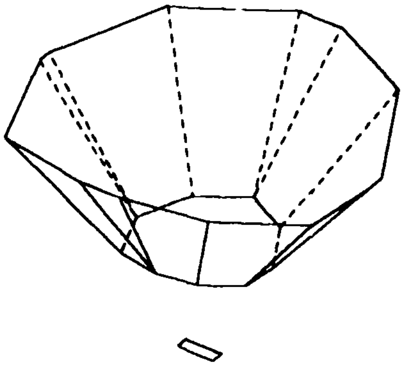


Figure 2. The computed occluded region of B.

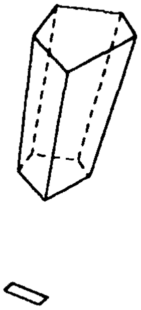


Figure 3. Viewing region for H.

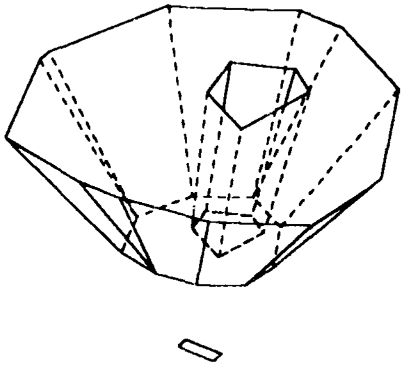


Figure 4. The resultant occluded region.

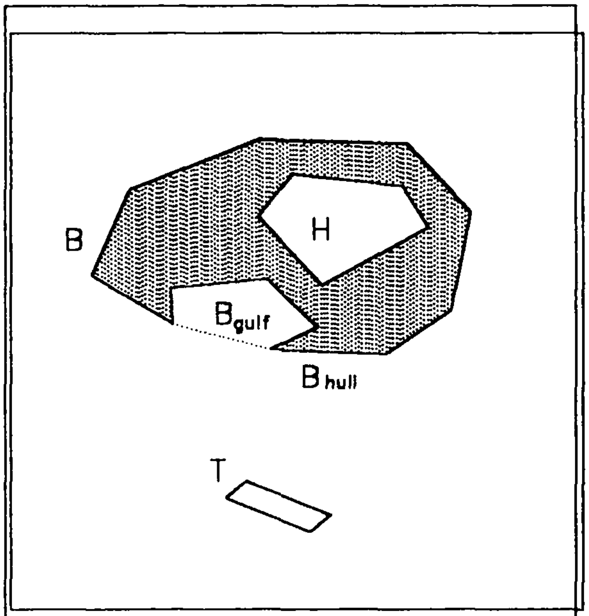


Figure 5. The occluding object is a concave polygon with a hole in it.

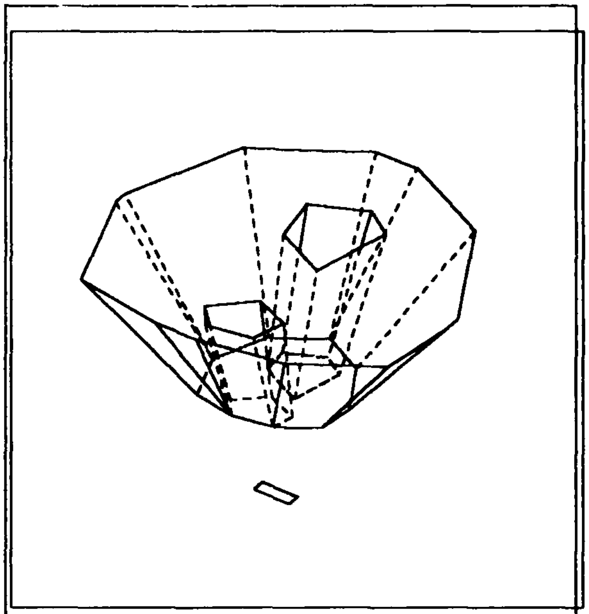


Figure 6. Incorrect occluded region if the gulf is not extended into an equivalent or virtual hole.

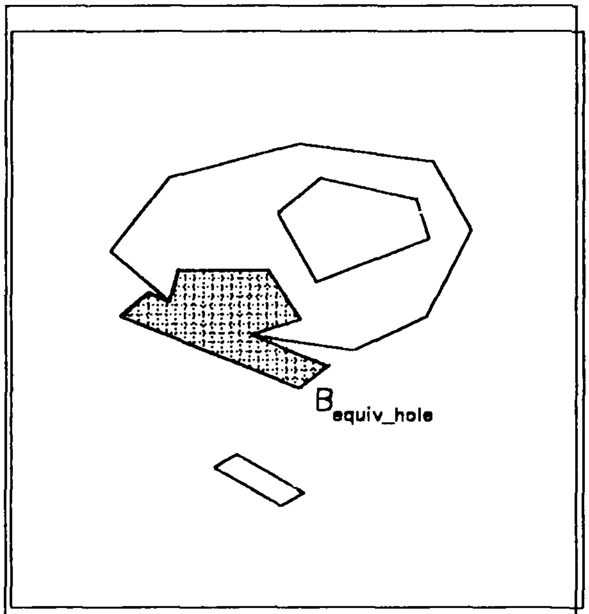


Figure 7. The gulf is extended to become a virtual gulf or equivalent hole.

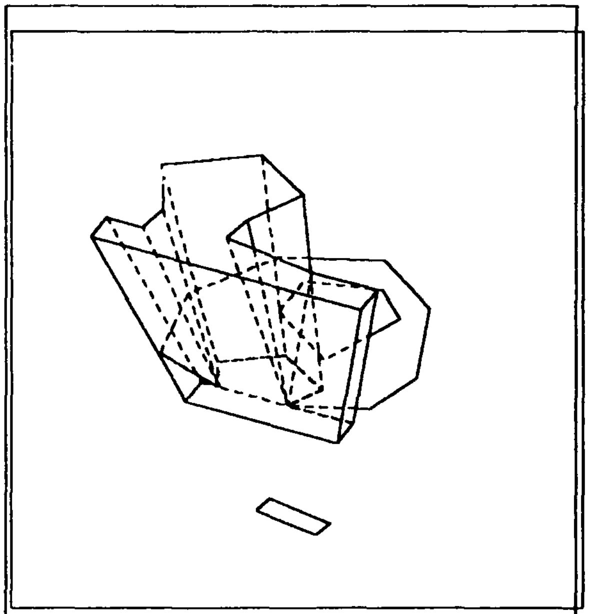


Figure 8. The viewing region through the equivalent hole.

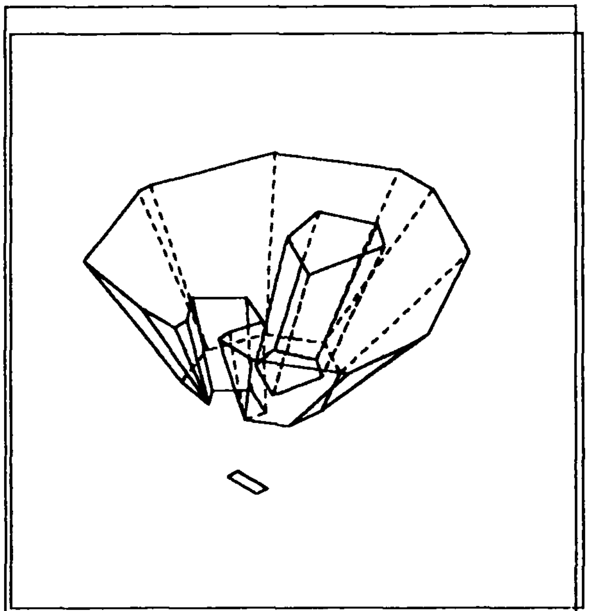


Figure 9. The resultant occluded region.

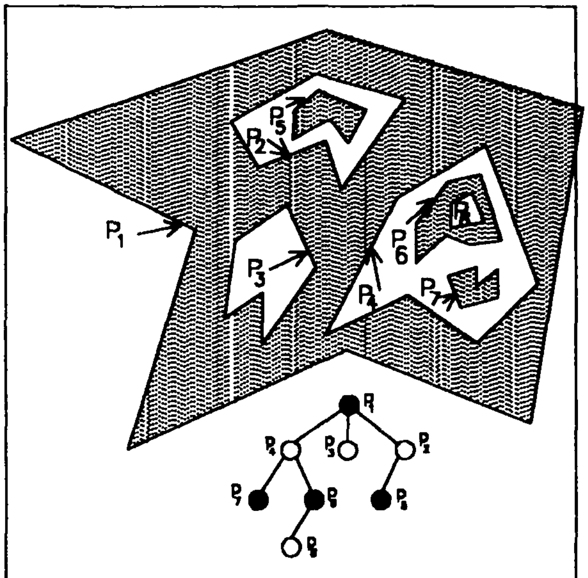


Figure 10. A polygon with loops nested within it and its loop tree.

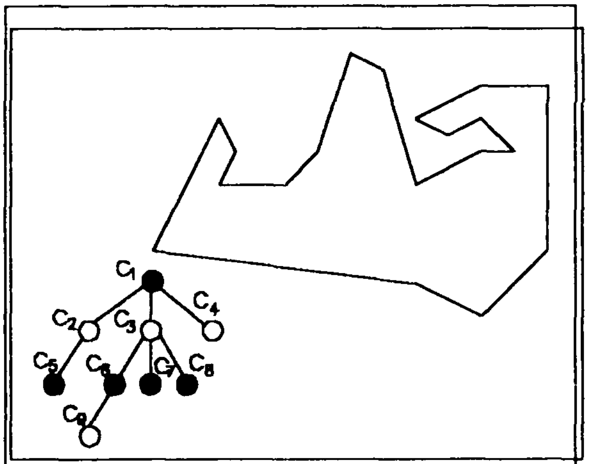


Figure 11. Concave polygon and its convex tree.

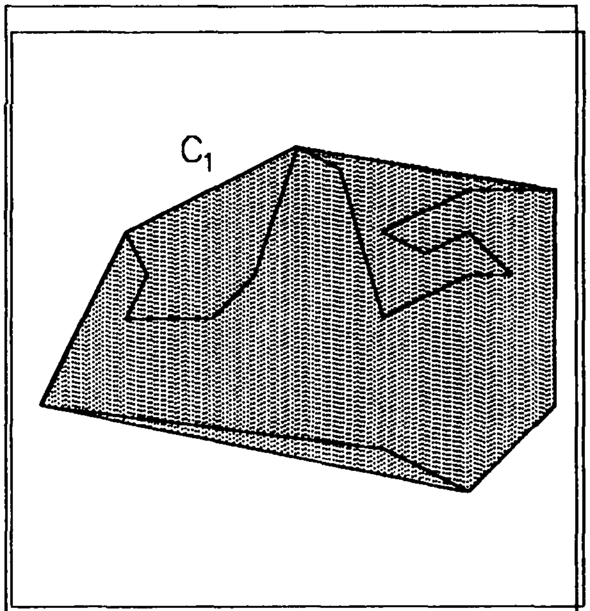


Figure 12. First level of the convex decomposition (convex hull).

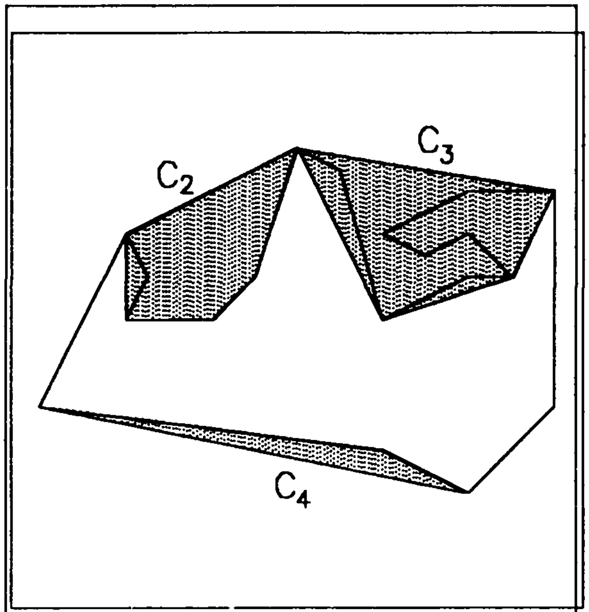


Figure 13. Second level of the convex decomposition.

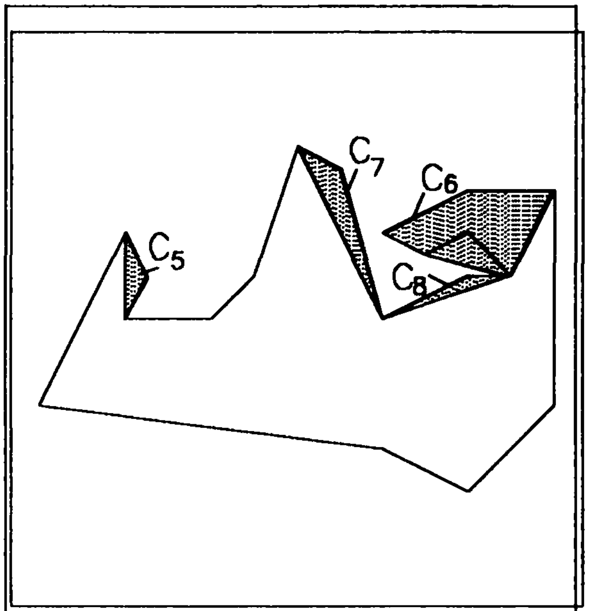


Figure 14. Third level of the convex decomposition.

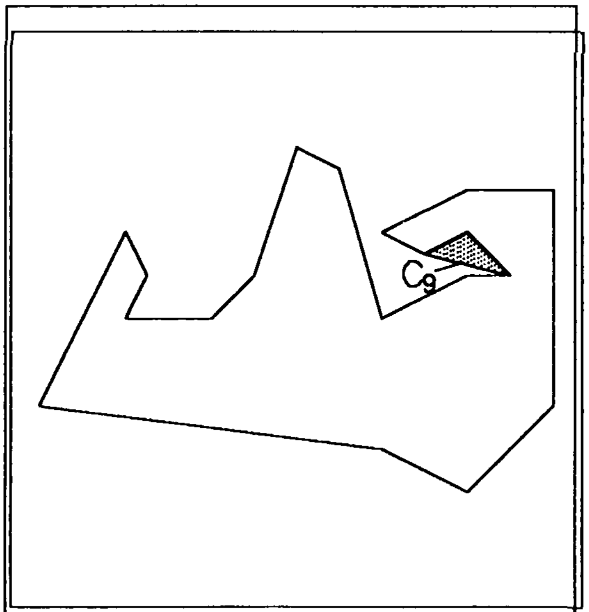


Figure 15. Fourth level of the convex decomposition.

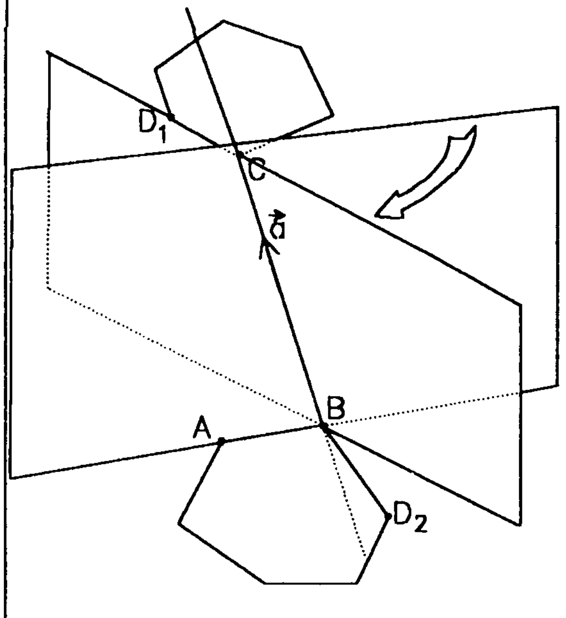


Figure 16. The Rolling method.

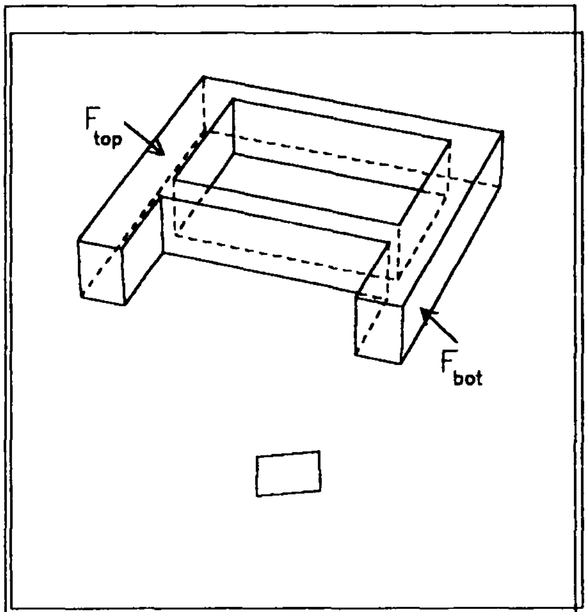


Figure 17. Polyhedron example.

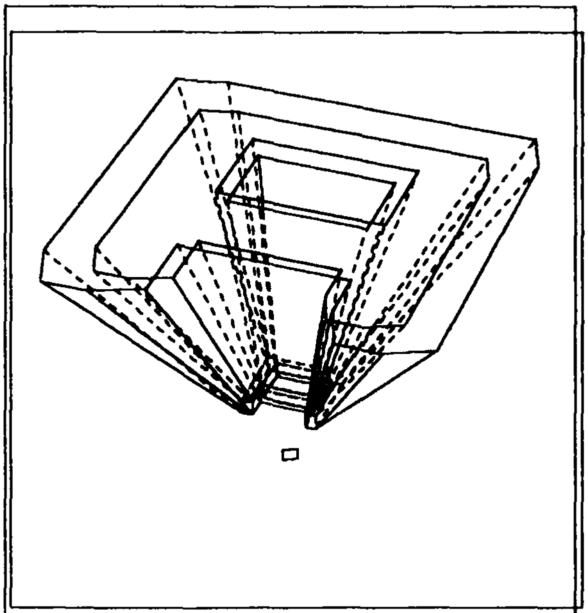


Figure 18. The two occluded regions superimposed.

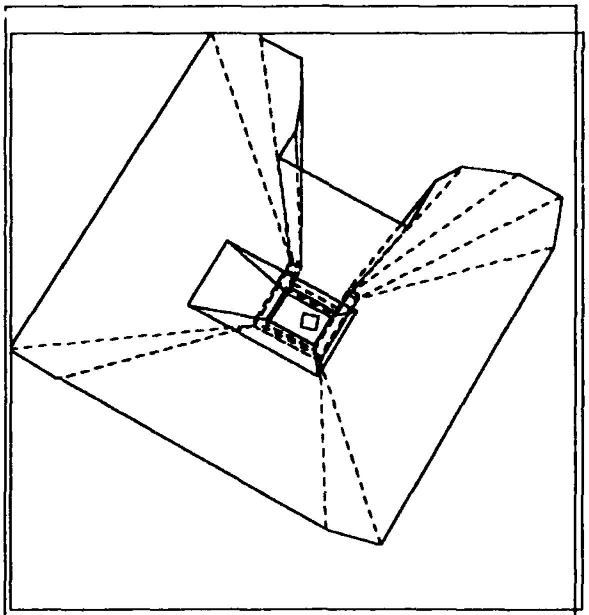


Figure 19. Non-critical viewpoint through the hole.

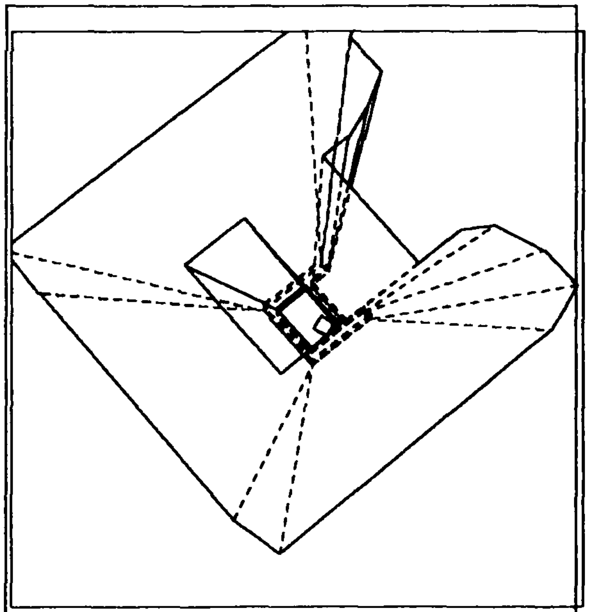


Figure 20. Critical viewpoint through the hole to validate accuracy of boundary of the constructed viewing region

REASONING ABOUT NONLINEAR INEQUALITY CONSTRAINTS: A MULTI-LEVEL APPROACH *

David Cyrluk
General Electric Company
Corporate Research and Development
Schenectady, NY 12345
cyrluk@ge-crd.arpa

Deepak Kapur
Department of Computer Science
State University of New York at Albany
Albany, NY 12222
kapur@albanycs.albany.edu

ABSTRACT

A multi-level approach for reasoning about nonlinear algebraic inequality constraints is proposed. The approach improves upon Brooks' extension of Bledsoe-Shostak's SUP-INF method. The approach involves abstracting a nonlinear problem to a linear problem and to a problem in qualitative reasoning about the signs of nonlinear terms. The results of these two abstractions of a nonlinear problem are used to refine bounds for nonlinear terms and finally, to compute bounds for variables. The approach is motivated by our work on using algebraic and geometric constraints in model-based vision. It is also likely to have applications in constraint logic programming and constraint-based languages.

MOTIVATION

Constraints arise naturally in many artificial intelligence applications including model-based vision [Sugihara, 1984; Brooks. 1981; Barry et al, 1988; Cyrluk et al, 1987; Cyrluk et al, 1988], simulation and graphics [Borning, 1979], solid and geometric modeling, robotics, design [de Kleer and Brown, 1984], qualitative reasoning and algebra [de Kleer and Brown, 1984; Williams, 1988], data bases, theorem proving and natural language understanding. Constraints are also being incorporated in logic programming languages to enhance their expressive power, especially for engineering design and other applications. Constraint solving is perhaps the most computationally intensive operation in constraint logic programming languages [Jaffar and Lassez, 1987; Heintz et al, 1986]. Constraint-based programming has been proposed as a new paradigm for programming which is especially found to be helpful for graphics applications [Leler, 1987].

This paper discusses a multi-level approach for reasoning about nonlinear algebraic inequality constraints. Our work is motivated by the use of these constraints in model-based vision, constraint logic programming, and reasoning about computations and their specifications. The problem under consideration is:

Given a finite set of nonlinear inequality constraints over variables ranging over the reals, find whether the constraints can be satisfied. If the answer is yes, then determine intervals of values which the variables must

*The work reported here was partially supported by the DARPA Strategic Computing Program under the Army Engineer Tcpographic Laboratories, Contract No. DACA76-86-C-0007, and by the Air Force Office of Scientific Research (AFSC), under Contract F49620-89-C-0033.

take to satisfy the constraints.

This problem falls in the theory of real closed fields and there exist many complete decision procedures for this theory and an interested reader may look at [Tarski, 1948; Arnon et al, 1984; Canny, 1987] for details. However, our experience is that these procedures are so general purpose that even simple problems which can be easily done using heuristics by hand, sometimes cannot be done using these procedures in a reasonable amount of computer time and space (see [Davenport et al, 1988] for examples of such problems).

Our goal is not to develop a method that solves the above problem completely, since we suspect a complete method is very likely to be extremely inefficient. Instead, we are interested in a method which can *satisfactorily* solve significant instances of this problem arising in the above-mentioned application domains. In particular, we would like the method to be *sound*; that is, if the method declares that the constraints are unsatisfiable, then they indeed should be unsatisfiable. However, the method may not be able to determine unsatisfiability in all cases. In cases where the method does not find the constraints to be unsatisfiable, we would like it to (i) compute the intervals of possible values which individual variables must take for the constraints to be (possibly) satisfied, and (ii) generate additional information in the form of bounds on terms and polynomials which can be used incrementally, i.e., to refine these intervals for individual variables when additional constraints are imposed.

We propose a multi-level approach for reasoning about nonlinear constraints. The approach involves (i) abstracting a nonlinear problem to a linear problem, (ii) qualitative reasoning, (iii) propagating bounds on nonlinear terms, and (iv) solving nonlinear constraints. If the unsatisfiability of constraints is detected in any of these steps, the original problem is then unsatisfiable.

By considering each term in a nonlinear problem as a distinct variable and ignoring any relationship between nonlinear terms, nonlinear constraints can be abstracted as linear constraints. Linear constraints can be solved using a number of methods including methods for linear progamming or other methods proposed for this problem in the automated reasoning and program verification literature [Bledsoe, 1975; Shostak, 1977]. Nonlinear terms can now be further abstracted by considering their signs instead of their values. This gives rise to the following subproblem:

From the sign constraints on nonlinear terms, decide the sign of an arbitrary nonlinear term.

A similar problem arises in qualitative reasoning [Williams, 1988]. Once the sign information about arbitrary nonlinear terms is available, it can be used to obtain tighter bounds on subterms including the individual variables.

We discuss below how nonlinear constraints arise in different application domains. We give an overview of Brooks' extension of the SUP-INF method for reasoning about nonlinear constraints. We discuss the limitations of the method as well as improvements made to the method to enhance its applicability. Finally we discuss the multi-level approach to reasoning about nonlinear constraints, and illustrate it using an example.

APPLICATIONS

The need for reasoning about nonlinear constraints arises in different aspects of machine vision, qualitative reasoning, reasoning about computations and specifications, and constraint programming.

Many aspects of machine vision require manipulating algebraic constraints, in particular, for generating models from descriptions, for matching a parameterized model specified as a set of constraints against an image with errors, as well as for recovering three dimensional topological information about objects. Brooks and Sugihara pioneered the approach of using algebrain constraints in machine vision. Sugihara [Sugihara, 1984] showed how labeling line drawings of polyhedral objects can be interpreted as solving linear constraints. Brooks [Brooks, 1981] demonstrated the use of his extension of the SUP-INF method for matching parameterized models based on generalized cylinders. He also illustrated how prediction and error correction can be performed by constraint solving.

In [Cyrluk et al, 1987] we began investigating formal approaches to model formation and model matching using algebraic and geometric reasoning methods. In particular, we defined the view consistency problem, which uses geometric constraints deduced from an image for matching against another image. Nonlinear constraints (equalities and inequalities) naturally arise as projection constraints, constraints due to vertices belonging to a common face or surface, parameterization constraints and constraints introduced to model uncertainity and errors. The work reported in this paper is motivated by these applications; preliminary investigations were described in [Cyrluk et al, 1987] as well as in [Barry et al, 1988].

In qualitative reasoning and its use in diagnostics and design one is often not interested in quantitative values but more in the nature of values such as whether they are positive, negative, increasing, or decreasing. Given qualitative information about certain quantities, one is interested in deducing qualitative information about other quantities which are useful in making proper design decisions. The multi-level approach presented in the paper employs an efficient qualitative reasoning algorithm to decide the sign problem of nonlinear terms from the signs of a given set of nonlinear terms.

Nonlinear constraints also arise while reasoning about the behavior of hardware and software. An example is checking whether an index in an array falls within its bounds. In fact, many algorithms for deciding the satisfiability of formulas over the integers (Pressburger arithmetic) were motivated by their application in reasoning about programs and specifications [Bledsoe, 1975; Shostak, 1977] An interested reader may consult these papers for details.

In the application of constraint programming in the logical or equational paradigm, nonlinear algebraic constraints arise while modeling graphics examples as well as solving engineering problems and other applications, including option trading [Heintz et al, 1986; Jaffar and Lassez, 1987]. Incremental algorithms for solving nonlinear constraints are needed. The reader may consult [Heintz et al, 1986] as well as [Jaffar and Lassez, 1987] for a good discussion of properties that such constraint solving algorithm must satisfy.

SUP-INF METHOD AND ITS EXTENSION

As stated above, the problem of solving nonlinear constraints falls within the theory of real closed fields. This theory was shown to be decidable by Tarski in the 1930's. Since then there has been considerable research done in improving his decision procedure. In particular, the work of Collins and his students, and Canny are worth mentioning. There exists an elegant implementation of Collins' *cylindrical algebraic decomposition* method in the computer algebra system SAC-2. We are, however, not aware of any implementation of these uniform decision procedures which can be satisfactorily used for solving nonlinear constraints arising in application domains discussed above in particular machine vision. This is due to the large number of variables in constraints for these applications.

An example of a particularly interesting incomplete procedure for solving nonlinear constraints is Brooks' adaptation of the SUP-INF method for solving linear constraints developed by Bledsoe and modified by Shostak. The original SUP-INF method was developed for deciding the unsatisfiability of linear constraints over the integers. Brooks adapted the SUP-INF method to nonlinear constraints including trigonometric functions to be used for parameterized model matching in a model-based vision system ACRONYM. He claimed that this extension worked quite well on the examples arising in his application. Others have also reported the use of Brooks' method for solving algebraic and geometric constraints arising in image understanding applications [Fisher and Orr, 1987]. A hierarchical approach to reasoning about inequality constraints which combines different methods was proposed in [Sacks, 1987]. Some of the limitations of Brooks' method were pointed out in that paper.

Below, we first give an overview of the SUP-INF method. We then discuss Brooks' extension for considering nonlinear constraints and discuss our findings regarding the limitation of this method and modifications to the method which significantly improved the performance of the method.

BROOKS' EXTENSION OF THE SUP-INF METHOD

The input to the SUP-INF method is a conjunction of linear inequalities with rational (or integer) coefficients (an equality constraint $p_1 = 0$ is transformed into a conjunction of two inequalities $p_1 \leq 0$ and $p_1 \geq 0$). The goal is to decide whether these inequalities can be satisfied. If satisfiable, the method produces lower and upper bounds for each variable appearing in the set of inequalities. A set of inequalities is unsatisfiable if and only if there is a variable for which its lower bound is greater than its upper bound.

The method is based on transforming inequalities such that each variable x can be expressed as $x \leq ub_i$ or $x \geq lb_i$, where ub_i and lb_i are, in general, linear expressions in terms of the rest of the variables. An upper bound for x, SUP(x), is then the minimum over ub_i's, whereas a lower bound for x, INF(x), is the maximum over lb_i. To compute the minimum of ub_i's, say, the algorithm is called recursively on each ub_i in an attempt to compute the lower and upper bounds on ub_i in terms of the variable x. Finally, linear equations in terms of x are obtained for these bounds, which can be solved. A dual technique is used for computing the lower bounds. In this way, the algorithm computes rational upper and lower bounds for each variable.

Shostak proved that these bounds are tight, and improved this method for deciding satisfiability over the integers. If the inequalities do not have a rational solution, then they do not have an integer solution either. However, if they do have a rational solution, i.e., if the method produces satisfiable intervals of upper and lower bounds for each variable, it can be checked whether each interval has an integer. If for some variable, its interval does not include an integer, the inequalities do not have an integer solution. Otherwise, the procedure has to search over integers in the satisfiable interval of each variable.

Brooks extended Bledsoe-Shostak's SUP-INF method to be applicable to nonlinear inequalities. The basic approach is the same as in the linear case. For each variable x, inequalities are transformed so that all terms in which x appears have lower bounds lb_i's and upper bounds ub_i's expressed in terms of other variables. The variable x is then factored out and lb_i's and ub_i's are divided by appropriate polynomials to compute a lower bound as the maximum over rational functions and an upper bound as the minumum over rational functions (a rational function is a polynomial divided by another polynomial). The main distinction is that before dividing a polynomial by another polynomial, the extended algorithm performs qualitative reasoning to determine the sign of the polynomial used as a divisor. For that, the method attempts to determine the sign of variables - whether the value is always positive, zero or negative. Sign constraints from nonlinear terms are propagated to generate sign constraints on variables, which in turn, constrain the signs of other nonlinear terms involving these variables. For instance, if the sign of xy is known to be positive, then x and y will have the same sign, either both are positive or both are negative.

Modifications

Brooks' method is limited for a number of reasons. The method deals with each nonlinear term separately without constraining other terms in which common variables appear. For instance, as pointed out by [Sacks, 1987], for computing an upper bound of $x^2 - x$ when x is unconstrained, an upper bound for x^2 and a lower bound for x are computed independently and the same value of x is not used for determining bounds of $x^2 - x$.

Since Brooks had found his method to be quite useful for parameterized model matching, we implemented Brooks' extended SUP-INF method in *GEOMETER*, an algebraic and geometric reasoning system for image understanding applications [Harris et al, 1988; Barry et al, 1988]. We experimented with it on a number of examples including examples from model matching and model formation. We tried the method on simple examples arising in parameterized model matching. Our experience with the method was not positive. We then analyzed Brooks' method and investigated the following heuristics to improve its performance:

1. Handling equational constraints by simplification using the Gröbner basis algorithm.

2. Storing the results of intermediate computations.

3. Identifying a subset of variables as parameters.

4. Propagating sign information about variables by multiple runs of the method.

5. Computing disjoint intervals for variables by case analysis on the signs of variables.

Below, we briefly review each of the improvements; details, with examples can be found in [Cyrluk et al, 1988].

One of the first modifications was to combine Brooks' extended SUP-INF method with the Gröbner basis algorithm [Buchberger, 1985], which is already implemented in GEOMETER [Harris et al, 1988]. The basic idea is to first manipulate equality constraints using the Gröbner basis algorithm, possibly deducing additional equality constraints. Equality constraints are then used as rewrite rules to simplify the inequality constraints. All constraints (input as well as deduced) are then transformed into inequalities and the SUP-INF method is invoked. If any new equality constraint is detected (when the satisfiable interval for some variable or term includes only one value, i.e., its upper bound and lower bound are identical), then that equality constraint is further propagated using the Gröbner basis algorithm. This improved the performance since the worst-case complexity of Bledsoe-Shostak's method is exponential in the number of variables and handling equality constraints using rewriting techniques can reduce the number of variables needed to be considered.

Another modification made to Brooks' method is to specify a subset of the variables as input variables. When SUP and INF are called on these variables, no extra computation takes place, rather the user provided bounds are used. This modification is especially useful in parameterized model matching problems where there are a large number of variables involved [Cyrluk et al, 1988].

The SUP-INF method is highly recursive. We observed that many computations in the method were being repeated, so storing intermediate computations improved the performance, i.e., SUP and INF of a variable in terms of other variables can be stored for future use. This memory feature, which Brooks also mentioned in his paper, improved the performance of the algorithm by two orders of magnitude on some nontrivial examples; see [Cyrluk et al, 1988] for details.

The above modifications improved the performance of the method significantly. Despite this, the method still was not good enough to be useful for large examples with more than 20 variables; further, most of these variables are in almost every constraint. The main reason for the poor performance is the highly recursive nature of the SUP-INF method. The interaction graph of the subproblems considered in the recursive calls indicates that in the worst case there are $O(n^n)$ subproblems being generated. By identifying redundant computations, it is possible to ensure that all the constraints are taken into account by generating $O(n^3)$ subproblems. Note that the algorithm is still exponential due to the symbolic addition and subtraction of terms involving min and max. Handling rational functions of the form $max(p_1, \cdots, p_k)/min(q_1 \cdots, q_l)$, where p_i, q_i are nonlinear polynomials, arising in the extension for the nonlinear case made the performance worse.

Brooks had observed in his paper, "in practice we have not encountered any case where the method failed to detect any inconsistency." However, he could not identify what class of problems could be solved by his method. We identified simple examples of unsatisfiable nonlinear inequalities which required more than one run of the method [Cyrluk et al, 1988]. It is possible to design a family of examples which require arbitrary many runs to detect inconsistency. Further, we were able to design examples of satisfiable inequalities such that the bounds on variables could be improved by multiple runs. This phenomenon has to do with the inability to compute the sign information about variables and terms in a single run since variables are considered in the method in some order.

For linear inequalities, Shostak proved that the SUP-INF method computes tight lowe, .nd upper bounds for variables. But for nonlinear inequalities, the extended SUP-INF method may not generate good bounds in a single run; in fact, there are examples for which better bounds exist (in the form of more than one interval) but the SUP-INF method cannot find them [Cyrluk et al, 1988]. One possible way to deal with this limitation is to perform case analysis on the possible values of selected variables, i.e., whether a variable is zero, positive or negative. This modification resulted in decomposing the original problem into many subproblems and gave better (but disjoint) intervals for variables.

A MULTI-LEVEL APPROACH

In this approach, the satisfiability of nonlinear algebraic constraints is not directly determined. Rather, the problem is abstracted at two different levels. It is assumed that the equality constraints have already been processed using the Gröbner basis algorithm as explained above and that they have been tranformed into a conjunction of inequality constraints. Initially, every nonlinear term appearing in the problem is abstracted to be a distinct variable. Thus nonlinear terms with common variables will be abstracted to distinct variables as though there is no relation between them. This transformation abstracts the nonlinear problem to a linear problem, which can be solved using well known techniques for solving linear constraints. If the linear problem is not satisfiable then the nonlinear problem is also not satisfiable, but not necessarily vice versa.

Using the solutions obtained from the linear problem, which are intervals of values each nonlinear term can take, we now perform a qualitative abstraction and analyze the relationship between various nonlinear terms by examining their signs. If these signs are not consistent, then the original problem is unsatisfiable. Otherwise, the sign information is used to further refine the crude bounds obtained from the first level. This also can be done using linear programming techniques. Finally, Brooks' extension of the SUP-INF method can be used for getting better bounds using the sign information obtained from an earlier step.

Here is a top-level description of our heuristic procedure.

Input: C: A set of nonlinear constraints.
Output: B: Bounds on the variables appearing in C including bounds on nonlinear terms as well as polynomials.

Step 1 (Variable Abstraction): C_{VA} := *variable_abstract*(C). Treat each distinct nonlinear term as a new distinct variable.
Step 2 (Solving Linear Constraints): If C_{VA} is inconsistent, then C is inconsistent. Otherwise, B_{VA} := Bounds on the variables in C_{VA} (the terms in C). If t_i is a term in C, then B_{VA} will consist of bounds of the form $l_i \leq t_i \leq u_i$.
Step 3 (Sign Reasoning): From the bounds of the nonlinear terms in B_{VA}, get their signs and check for their consistency. If signs are inconsistent, then C is inconsistent. If signs are consistent, then use these signs to deduce the signs of arbitrary subterms needed in the following steps.
Step 4 (Bound Propagation): From the signs in step 3, refine the bounds obtained in step 2 and compute bounds on the original variables appearing in C.
Step 5 (Further Refinement): Use techniques for reasoning about nonlinear constraints to further refine these bounds.

VARIABLE ABSTRACTION: TRANSFORMING NONLINEAR PROBLEMS TO LINEAR PROBLEMS

This step is straightforward. Every nonlinear term is replaced by a distinct variable. This transforms a nonlinear problem into a linear problem. As an example consider the following constraints:

$$\begin{aligned}
2uv^3xy^2 + vx^5yz^4 &\geq 1\\
2uv^3xy^2 - 3vx^5yz^4 &\geq 1\\
-uv^3xy^2 + 2vx^5yz^4 - uz - 4uxy &\geq -6\\
-3uv^3xy^2 + vx^5yz^4 + 3uz + 5uxy &\geq -8
\end{aligned}$$

After variable abstraction, the following linear constraints are obtained:

$$\begin{aligned} 2t_1 + t_2 &\geq 1 \\ -4t_1 - 3t_2 &\geq 1 \\ -t_1 + 2t_2 - t_3 - 4t_4 &\geq -6 \\ -3t_1 + t_2 + 3t_3 + 5t_4 &\geq -8 \end{aligned}$$

Solving Linear Constraints

Given the set of linear constraints of the form $\sum_i (c_i * t_i) \geq c$, where the c_i and c are rational numbers, and t_i are real valued variables; determine whether the set of constraints is satisfiable, and if so find *tight* bounds: $l_i \leq t_i \leq u_i$. By the following proposition, if the linear problem is unsatisfiable, then the original nonlinear problem is unsatisfiable, but the converse does not hold. The bounds obtained in this step are usually cruder, because the relationship between different nonlinear terms is not considered. The remaining steps attempt to refine these bounds further.

Proposition 1: Given a set of nonlinear constraints C, if *variable_abstract*(C) is unsatisfiable, then C is unsatisfiable.

We have explored the use of three methods: (i) linear SUP-INF, (ii) a modification to the linear SUP-INF in which equality constraints are handled using rewriting techniques [Buchberger, 1985; Cyrluk et al, 1988], and (iii) the Simplex linear programming algorithm. For large problems such as those arising in in the model formation and parameterized model matching applications of machine vision, the simplex method performed the best because the performance of the SUP-INF method and its modification gets worse as the number of variables in a problem increases.

In the example above linear programming produces the following bounds:

$$t_1 \geq 2 \qquad t_2 \leq -3 \qquad t_3 \leq -2 \qquad t_4 \geq 1$$

If a variable has the same lower and upper bound implying that its value is uniquely determined, that variable can be eliminated by uniformly substituting its value.

QUALITATIVE REASONING: SIGN CONSISTENCY AND COMPUTING SIGNS

From the crude bounds on nonlinear terms obtained in the previous step, we abstract the values to their signs. We would like to decide whether the constraints in C can restrict each term (including the original variables in C) to one of the following five signs: zero, positive, negative, nonnegative (only if the term can have 0 as well as positive values), and nonpositive (only if the term can have 0 as well as negative values).[1] For the case when a term lies within the range of a negative and positive real, it is said to have the *unknown* sign. By the following proposition, if the signs of terms are not consistent, then the original set C of nonlinear constraints is not satisfiable.

Proposition 2: Let $\{l_i \leq \prod_{j=1}^{n} x_j^{e_j} \leq u_i \mid 1 \leq i \leq m\}$ be the bounds obtained by solving *variable_abstract*(C). If the terms in $\{\prod_{j=1}^{n} x_j^{e_j} \mid 1 \leq i \leq m\}$ cannot be consistently assigned unique signs (i.e., zero, positive, negative, nonnegative, nonpositive, unknown), then C is unsatisfiable.

The sign consistency check is done in two parts. First, we assume that none of the variables is zero, and determine whether terms can be consistently assigned positive and negative signs under that assumption. If not, there are two possibilities: there are terms whose value is zero, or the sign constraints are inconsistent

[1] A term that takes the value 0 is assumed to have the sign zero, and not both the nonnegative and nonpositive signs.

thus implying that the original set of constraints is inconsistent. To disambiguate these two cases, another qualitative reasoning check for determining whether terms can take zero values is performed.

Positive and Negative signs

Assuming that all variables are nonzero we give a polynomial time decision procedure for deciding whether terms can have negative or positive signs and if so computing their signs. This decision procedure can be used to deduce, for any nonlinear term involving variables which satisfy the sign constraints, whether the term has the negative or positive sign. The key observation is that multiplication on reals when applied to positive and negative signs behaves exactly as the boolean equivalence connective $\Leftrightarrow$. This observation allows us to transform the sign constraint problem to solving linear boolean equations over a boolean ring (or the field Z_2), in which the addition operator is the exclusive-or connective. We identify the negative sign with 0 (false) and the positive sign with 1 (true).

Corresponding to each variable x in C, we introduce a corresponding boolean variable, s_x, which has the value 0 (1) iff x is negative (positive). Given a term $t = \prod_{i=1}^{n} x_i^{e_i}$, we generate a propositional formula, *sign_prop*(t) using $\Leftrightarrow$ as follows:
1. remove all variables with even exponents.
2. make all odd exponents one.
3. replace $*$ with $\Leftrightarrow$ and x_i with s_{x_i}.
The $\Leftrightarrow$ connective in a propositional constraint can be converted into $\oplus$ (exclusive-or connective).

For example, *sign_prop*(uv^3xy^2) is $s_u \Leftrightarrow (s_v \Leftrightarrow s_x)$.

For constraints of the form $l_i \leq t_i \leq u_i$ we generate sign constraints as follows:

1. If $l_i \geq 0$ then generate *sign_prop*$(t_i) = 1$.
2. If $u_i \leq 0$ then generate *sign_prop*$(t_i) = 0$.
3. Otherwise the sign of t_i is unknown and no sign constraint is generated.

Note that this translation works because we are assuming that the signs of all the variables are nonzero.

For example, $uv^3xy^2 \geq 0$ is equivalent to $(s_u \Leftrightarrow (s_v \Leftrightarrow s_x)) = 1$. When $\Leftrightarrow$ is changed to $\oplus$, we get $s_u \oplus s_v \oplus s_x = 1$.

Proposition 3: Assuming that all the x_i are nonzero, the sign of a term $t = \prod_{i=1}^{n} x_i^{e_i}$ is positive (negative) iff *sign_prop*(t) is 1(0).

Given $\{l_i \leq \prod_{j=1}^{n} x_j^{e_j} \leq u_i \mid 1 \leq i \leq m\}$, for every positive or negative term, associate a proposition constructed using *sign_prop* with it. We obtain a finite set (in general $< m$) of boolean equations. These boolean equations can be solved using Gaussian elimination (an n^3 algorithm where n is the number of boolean variables) or more efficient algorithms based on matrix multiplication and inversion [Baase, 1978].

From the constraints for the above example: $uv^3xy^2 \geq 2$, $vx^5yz^4 \leq -3$, $uz \leq -2$, and $uyx \geq 1$, the following boolean equations are obtained: $s_u \oplus s_v \oplus s_x = 1$, $s_v \oplus s_x \oplus s_y = 0$, $s_u \oplus s_z = 1$, $s_u \oplus s_y \oplus s_x = 1$. Using Gaussian elimination to generate a triangular system gives: $s_y \oplus s_z = 0$, $s_x = 0$, $s_v \oplus s_z = 0$, $s_u \oplus s_z = 1$.

Theorem 4: Let PC be the set of propositional constraints generated from $\{l_i \leq \prod_{j=1}^{n} x_j^{e_j} \leq u_i \mid 1 \leq i \leq m\}$. A term t is positive (negative) if and only if $PC \models$ *sign_prop*$(t) = 1$ (or $PC \models$ *sign_prop*$(t) = 0$).

An n^3 algorithmm can also be obtained by converting equations into rewrite rules. Note that the xor operator, $\oplus$, satisfies the equations $x \oplus x = 0$. By converting equations into rewrite rules using an ordering

$s_y > s_x > s_v > s_u > s_z$, we get: $s_x \rightarrow s_u \oplus s_v \oplus 1$ from the first equation. Using this rule, the second equation simplifies to: $s_y \oplus s_u = 1$ which gives a rewrite rule $s_y \rightarrow s_u + 1$. The third equation gives the rewrite rule: $s_u \rightarrow s_z + 1$. These three rules simplify the fourth equation to: $s_v \oplus s_z = 0$ thus giving a rule $s_v \rightarrow s_z$. If the rules are interreduced, then we get a reduced set of rewrite rules: $s_y \rightarrow s_z$, $s_x \rightarrow 0$, $s_v \rightarrow s_z$, $s_u \rightarrow s_z \oplus 1$, which are the rules corresponding to the reduced triangular system generated by Gaussian elimination. These rewrite rules can be used as a decision procedure to determine the sign of arbitrary terms. For example, given the term uv, $sign\_prop(uv) = s_u \oplus s_v \oplus 1$, which can be rewritten using these rules to: $s_u \oplus s_v \oplus 1 \rightarrow s_u \oplus s_z \oplus 1 \rightarrow s_z \oplus s_z \oplus 1 \oplus 1 = 0$, thus indicating that uv is negative.

If the sign constraints are satisfiable, then the sign information can be used in steps 4 and 5 to refine the bounds on terms. If the sign constraints are unsatisfiable, this inconsistency could be because a term is 0, or the original constraints are unsatisfiable. To check for the first case of terms being 0, a different sign constraint satisfaction problem in which terms can have either 0 or nonzero values, is solved.

Zero and Non-zero Values

From the above sign reasoning step, we have additional information that the product of all the variables appearing in the constraints is 0, which we add as an additional constraint. From the intervals obtained for terms by solving the linear constraint problem, it is known what variables (and hence terms) cannot be 0, i.e., if a term t is < 0 or > 0, then each of the variables in t must be nonzero. From each constrained term which has a nonnegative, nonpositive or an unknown sign, the nonzero variables can be eliminated without changing their qualitative value being zero or nonzero. After this elimination, if there are certain subterms left which could possibly take 0 as their values, then the sign consistency check succeeds. This check can be done in linear time.

The qualitative information thus obtained is not necessarily complete. For instance the above algorithm may say that $xyzu = 0$ meaning that at least one of the variables x, y, z and u is 0. However it might be the case that both $xy = 0$ and $zu = 0$ can be deduced. Minimal terms that are 0 can be computed using an exponential time algorithm.

Relationship to Williams' Qualitative Algebra

Williams [Williams, 1988] also needs to reason about signs in a qualitative reasoning application. The main difference between his method and ours is that we restrict our sign algebra to deal only with multiplication. This allows us to come up with an efficient decision procedure to decide the sign problem, whereas his qualitative sign algebra in general cannot be solved. The purpose of our two systems is vastly different. Williams' ultimate goal is to reason about thresholds. Ours is to reason about signs to aid us in deriving better numerical bounds.

PROPAGATING BOUNDS ON TERMS USING SIGN INFORMATION

Given a set B_{VA} of bounds on terms of the form $l_i \leq t_i \leq u_i$, where $t_i = \prod_{j=1}^{n} x_j^{e_j}$, find optimal bounds : $l_j \leq x_j \leq u_j$

This problem is *almost* the dual of the linear problem of step 2 (with multiplication replacing addition). The difference is that when dividing, the sign of variables must be taken into account, whereas when subtracting, it does not have to be.

In linear algebra, given, $l_{12} \leq t_1 + t_2 \leq u_{12}$ and $l_1 \leq t_1 \leq u_1$ then $l_{12} - l_1 \leq T_2 \leq u_{12} - l_1$. But with products: given, $l_{12} \leq t_1 * t_2 \leq u_{12}$ and $l_1 \leq t_1 \leq u_1$ then

1. if t_1 is positive then $l_{12}/l_1 \leq t_2 \leq u_{12}/l_1$,
2. if t_1 is negative then $u_{12}/u_1 \leq t_2 \leq l_{12}/u_1$

Positive Variables When the sign of all the variables are positive simple modifications of the methods used in step 2 can be used, or the problem can be converted to one involving linear constraints. This can be done by taking the logarithm of the terms (products) to obtain sums. This is only defined when the signs of all the variables are positive. This is summarized in the following theorems.

Proposition 5: The function, $\phi : x \mapsto \log(x)$ is an isomorphism betweeen $(\Re^+, \{*, /, >, min, max\})$ and $(\Re, \{+, -, >, min, max\})$, where $\Re^+$ does not include 0.

Definition: Given the set B_{VA} in which the sign of every x_j is positive. Define $LOG(B_{VA})$ to be the set of linear constraints of the form, $\log(l_i) \leq S_i \leq \log(u_i)$, where $S_i = \sum_{j=1}^{n} e_j * \log\_x_j$, and $\log\_x_i$ are new variables.

Proposition 6: Let $l_i \leq x_i \leq u_i$ be the set of optimal bounds of B_{VA} (again with only positive variables), and let $ll_i \leq log\_x_i \leq lu_i$ be the set of optimal bounds for $LOG(B_{VA})$. Then $ll_i = \log(l_i)$ and $lu_i = \log(u_i)$.

Negative Variables When the signs of all the variables are determined but are not all positive, an equivalent set of constraints can be obtained in which they are all positive. For each negative variable, x_i, replace it with a positive variable, $x'_i = -x_i$, and modify the constraints accordingly.

Definition: Let B_{VA} be a set of bounds on terms where some of the variables might be negative, but the sign of all the variables are known. Define $POS(B_{VA})$ to be the set of constraints in which all the variables are positive, obtained from B_{VA} through the transformation described above.

For example the set of constraints:

$$\begin{array}{rcccl} -3 & \leq & x & \leq & -1 \\ 2 & \leq & y & \leq & 5 \\ -4 & \leq & xy & \leq & -1 \\ 3 & \leq & x^2y & \leq & 32 \end{array} \quad \text{is equivalent to:} \quad \begin{array}{rcccl} 1 & \leq & x' & \leq & 3 \\ 2 & \leq & y & \leq & 5 \\ 1 & \leq & x'y & \leq & 4 \\ 3 & \leq & x'^2y & \leq & 32 \end{array}$$

with $x' = -x$.

Proposition 8: Let x be a variable that is negative in B_{VA}, and x' be its corresponding variable in $POS(B_{VA})$. If $l \leq x' \leq u$ is the optimal bound for x' in $POS(B_{VA})$, then $-u \leq x \leq -l$ is the optimal bound for x in B_{VA}.

Thus when the sign of all the variables are known the problem of propagating bounds through terms can be solved as a linear programming problem.

Unknown Signs When the sign of only a few variables are unknown then multiple linear programming problems can be generated by performing case analysis on the signs of those variables. After solving the linear programming problems the bounds on the original variables can be obtained by taking the antilog of the new bounds.

Definition: Let B_{VA} be a set of bounds on terms, but with variables $ux_1, \ldots, ux_k$ having unknown signs. Then define $CA(B_{VA})$ to be the set of bounds obtained from B_{VA} in which the sign of all the variables are known. $CA(B_{VA})$ consists of 3^k bounds and is obtained by considering all possible signs (positive, negative, and zero) of the k variables in B_{VA} with unknown signs.

Proposition 9: Let $l_i \leq x \leq u_i$ be the set of optimal bounds for x in each of the problems in $CA(BV_{VA})$. Then the optimal bound for x in B_{VA} is:

$$\min_i l_i \leq x \leq \max_i u_i.$$

FURTHER REFINEMENTS AND OTHER HEURISTICS

If an inconsistency of the constraints is not detected in the above steps, the result is lower and upper bounds for terms and variables. Additional heuristics can be used to refine these bounds further. In particular, if a univariate polynomial is generated during the computation, its roots can be found using numerical methods or by Sturm sequences [Davenport et al, 1988], which can be used to get better bounds on the variable and terms in which the variable appears. Finally, using the sign information about terms, Brooks' extension of the SUP-INF method can be used more effectively on polynomial inequalities.

REFERENCES

Arnon, D.S., Collins, G.E., and McCallum, S., "Cylindrical Algebraic Decomposition I, II," *SIAM J. of Computing* 13, 865-877; 878-889, 1984.

Baase, S., *Computer Algorithms: Introduction to Design and Analysis.* Addison-Wesley, 1978.

Barry, M., Cyrluk, D., Kapur, D., Mundy, J., Nguyen, V.D., "A Multi-level Geometric Reasoning System for Vision," *Proc. of an NSF Workshop on Geometric Reasoning*, Oxford, England, June 1986. Also in a special issue of the *Artificial Intelligence* Journal on *Geometric Reasoning* Dec. 1988.

Bledsoe, W.W., "A New Method for Proving Certain Pressburger Formulas," Advance Papers, *Fourth Intl. Joint Conf. on Artificial Intelligence,* Tibilisi, Russia, Sept. 1975, 15-21.

Borning, A., *THINGLAB: A Constraint-oriented Simulation Laboratory.* Stanford CS Report STAN-CS-79-746, July 1979.

Brooks, R.A., "Symbolic Reasoning Among 3D Models and 2D Images," *Artificial Intelligence* 17, 1981.

Buchberger, B "Gröbner Bases: An Algorithmic Method in Polynomial Ideal Theory," in: N.K. Bose (ed.) *Multidimensional Systems Theory*, Reidel, 184-232, 1985.

Canny, J.F., *The Complexity of Robot Motion Planning*, PhD. Thesis, Massachusetts Institute of Technology, May, 1987.

Cyrluk, D., Kapur, D., Mundy, J., and Nguyen, V., "Formation of Partial 3D Models from 2D Projections - An Application of Algebraic Reasoning," *1987 DARPA Image Understanding Workshop*, Feb. 1987, Los Angeles, Calif.

Cyrluk, D., Kapur, D., and Mundy, J., "Geometric and Algebraic Reasoning for View Consistency and Parameterzied Model Matching," *1988 DARPA Image Understanding Workshop*, April 1988, Cambridge, MA.

Davenport, J.H, Siret, Y., and Tournier, E., *Computer Algebra: Systems and Algorithms for Algebraic Computation.* Academic Press, 1988.

de Kleer, J., and Brown, J., "A Qualitative Physics Based on Confluences," *Artificial Intelligence* 24, 1984.

Fisher, R.B., and Orr, M.J.L., "Solving Geometric Constraints in a Parallel Network," Proc. of *1987 Alvey Vision Conference*, Cambridge, England, 1987.

Harris, R., Cyrluk, D., and Kapur, D., "GEOMETER: A Theorem Prover for Algebraic Geometry," Proc. of *Ninth International Conference on Automated Deduction*, Argonne, Illinois, May 1988.

Heintze, N.C., Jaffar, J., Lim, C.S., Michaylov, S., Stuckey, P.J., Yap, R., and Yee, C.N., "The CLP($\Re$) Programmer's Manual," *Monash University Technical Report No. 73*, June 1986.

Jaffar, J., and Lassez, J-L., "Constraint Logic Programming," Proc. of *Conference on Principles of Programming Languages*, Munich, Germany, 1987.

Leler, W., *Constraint Programming Languages: Their Specification and Generation.* Addison-Wesley, 1987.

Sacks, E.P., "Hierarchical Inequality Reasoning," Proc. *the National Conf. on Artificial Intelligence*, 1987.

Shostak, R., "On the SUP-INF Method for Proving Pressburger Formulas," *J. ACM* 24 (4), Oct. 1977.

Sugihara, K., "An Algebraic Approach to Shape From Image Problems," *Artificial Intelligence* 23, 1984.

Tarski, A., *A Decision Method for Elementary Algebra and Geometry*, University of California Press, Berkeley, 1948.

Williams, B., "MINIMA: A Symbolic Approach to Qualitative algebraic reasoning," Proc. of *AAAI 88*, St. Paul, Minnesota, 264-269, August 1988.

Measuring the Effectiveness of Task-Level Parallelism for High-Level Vision

Wilson Harvey, Dirk Kalp, Milind Tambe,
David McKeown, Allen Newell

School of Computer Science
Carnegie Mellon University
Pittsburgh, Pennsylvania
15213-3890

1. Abstract

Large production systems (rule-based systems) continue to suffer from extremely slow execution which limits their utility in practical applications as well as in research settings. Most efforts at speeding up these systems have focused on match or knowledge-search parallelism in production systems. Though good speed-ups have been achieved in this process, the total speed-up available from this source is not sufficient to alleviate the problem of slow execution in large-scale production system implementations. Such large-scale tasks can be expected to increase as researchers develop increasingly more competent rule-based systems.

In this paper, we focus on task-level parallelism, which is obtained by a high-level decomposition of the production system. Speed-ups obtained from task-level parallelism will multiply with the speed-ups obtained from match parallelism. Our vehicle for the investigation of task-level parallelism is SPAM, a high-level vision system, implemented in a production system architecture. SPAM is a mature research system having over 600 productions, with a typical scene analysis task having between 50,000 to 400,000 production firings and an execution time of the order of *10 to 100 cpu hours.*

We present a characterization of task-level parallelism in production systems and, from that, select an explicit, data-driven approach for exploiting task-level parallelism. We describe a methodology for applying the chosen approach to obtain a parallel task decomposition of SPAM and to arrive at our parallel implementation, SPAM/PSM. We present the results of that implementation that show near linear speed-ups of over 12 fold using 14 processors and that point the way to substantial speed-ups from task-level parallelism[1].

2. Introduction

Large production systems (rule-based systems) continue to suffer from extremely slow execution which limits their utility in practical applications as well as research settings. Most efforts at speeding up these systems have focused on match, i.e., knowledge-search, parallelism in production systems [3, 5, 7, 14, 19, 20]. Though good speed-ups have been achieved in this process, the total speed-up available from this source is limited. Therefore, match parallelism alone will not alleviate the problem of slow execution in production systems.

In this paper, we focus on task-level parallelism, which is obtained by a high-level decomposition of the production system. Speed-ups obtained from task-level parallelism will multiply with the speed-ups obtained from match parallelism. Our vehicle for the investigation of task-level parallelism is SPAM [11, 12, 13], a high-level vision system, implemented in a production system architecture. SPAM

[1] This research was partially supported by the Air Force Office of Scientific Research, under Grant AFOSR-89-0199, and by the Defense Advanced Research Projects Agency (DOD), ARPA Order No. 4976, monitored by the Air Force Avionics Laboratory Under Contract F33615-87-C-1499. The views and conclusions contained in this document are those of the authors and should not be interpreted as representing the official policies, either expressed or implied, of the Air Force Office of Scientific Research, of the Defense Advanced Research Projects Agency, or the US Government.

is a mature research system having over 600 productions, with a typical scene analysis task requiring between 50,000 to 400,000 production firings and an execution time of the order of 10 to 100 cpu hours[2]. Unlike most other production systems examined for studies in parallelism, it has embedded in it a large computational demand related to the vision task that it performs. This task-related computation is separate from the computation performed for knowledge-search in the system. This is evident in the large RHS processing time for this system. While many production systems spend up to 90% of their time in knowledge-search, SPAM spends only about 30-50% of its time there.

In this paper, we show that the opportunities for task-level parallelism in SPAM are high and provide a much larger payoff in speed-up than match parallelism. We present a methodology and a set of principles to arrive at a suitable parallel decomposition of the SPAM task that results in near linear speed-ups of over 12 fold using 14 processors on a 16-processor shared-memory multiprocessor. Our results also indicate that a potential speed-up of 50 to 100 fold may be achievable due to task-level parallelism. We further show that match parallelism, when used in conjunction with task-level parallelism, gives another multiplicative factor of speed-up which is proportional to the size of the match component in the overall execution time. In the SPAM system, this additional multiplicative factor is around 1.5 to 2.

This paper is organized as follows: Section 3 provides some background about production systems and SPAM, the image interpretation system that is the focus of our analysis of task-level parallelism. Section 4 discusses match parallelism and task-level parallelism in production systems. We describe a new organization to compare previous work in task-level parallelism along several independent dimensions. Section 5 discusses the implementation methodology used to determine appropriate levels for task-level parallelism. We also describe a set of experiments and measurements on SPAM that allowed us to select an appropriate grain of decomposition. These techniques should be applicable to the analysis of other large production systems for evaluating the opportunities for task-level parallelism.

A new system, SPAM/PSM, resulted from the apolication of this methodology and its implementation is described in Section 6. Section 7 presents a detailed analysis of the results of experiments across several dimensions including grain of decomposition, speed-ups due to processor allocation for match-level and task-level parallelism. Finally, Section 8 presents a summary of our research results and Section 9 discusses some issues for future work.

3. Background

In this section we provide a brief overview of OPS5 and SPAM. SPAM is implemented in OPS5, hence the description of OPS5 will be useful in understanding some of the issues in how SPAM represents knowledge about spatial and structural constraints used in computer vision. Besides providing background information, this section introduces the terminology that will be used in the rest of this paper.

3.1. OPS5

An OPS5 [2] production system is composed of a set of *if-then* rules, called *productions*, that make up the *production memory*, and a database of temporary data structures, called the *working memory*. The individual data structures are called working memory elements (WMEs), and are lists of attribute-value pairs. Each production consists of a conjunction of condition elements (CEs) corresponding to the *if* part of the rule (also called the left-hand side or LHS), and a set of actions corresponding to the *then* part of the rule (also called the right-hand side or RHS).

The CEs in a production consist of attribute-value tests, where some attributes may contain variables as values. The attribute-value tests of a CE must all be matched by a WME for the CE to match; the variables in the condition element may match any value, but if the variable occurs in more than one CE of a production, then all occurrences of the variable must match identical values. When all the CEs of a production are matched, the production is satisfied, and an instantiation of the production (a list of WMEs that matched it), is created and entered into the *conflict set*. The production system uses a selection procedure called *conflict-resolution* to choose a production from the conflict set, which is then

[2] These measurements are taken from the Lisp-based version of OPS5 running on a VAX/785 processor.

fired. When a production fires, the RHS actions associated with that production are executed. The RHS actions can add, remove or modify WMEs, or perform I/O.

The production system is executed by an interpreter that repeatedly cycles through three steps:

1. Match
2. Conflict-resolution
3. Act

The matching procedure determines the set of satisfied productions, the conflict-resolution procedure selects a single instantiation, and the act procedure executes its RHS. These three steps are collectively called the *recognize-act cycle.*

3.2. SPAM: A Production System Architecture For Scene Interpretation

SPAM [11, 12, 13] is a production system architecture for the interpretation of aerial imagery with applications to automated cartography and digital mapping. It tests the hypothesis that the interpretation of aerial imagery requires substantial knowledge about the scene under consideration. Knowledge about the type of scene — airport, suburban housing development, urban city — aids in low-level and intermediate level image analysis, and will drive high-level interpretation by constraining search for plausible consistent scene models. SPAM has been applied in two task areas: airport and suburban house scene analysis. In the remainder of this section we describe the SPAM architecture, and give run-time statistics that lead us to focus on one phase for our studies in parallelism.

As with many vision systems, SPAM attempts to interpret the 2-dimensional image of a 3-dimensional scene. A typical input image is shown in Figure 1. The particular goal of the SPAM system is to interpret an image segmentation, composed of image regions, as a collection of real-world objects. For example, the output for the image in Figure 1 would be a model of the airport scene, describing where the runway, taxiways, terminal-building(s), etc., are all located. SPAM uses four basic types of scene interpretation primitives: *regions*, *fragments*, *functional areas*, and *models*. SPAM performs scene interpretation by transforming image *regions* into scene *fragment* interpretations. It then aggregates these fragments into consistent and compatible collections called *functional areas*. Finally, it selects sets of functional areas to form *models* of the scene.

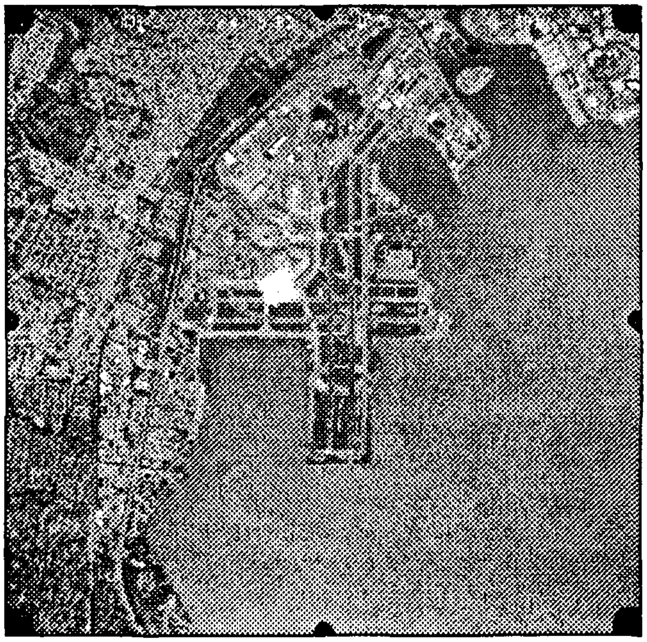


Figure 1: Aerial image of San Francisco Airport

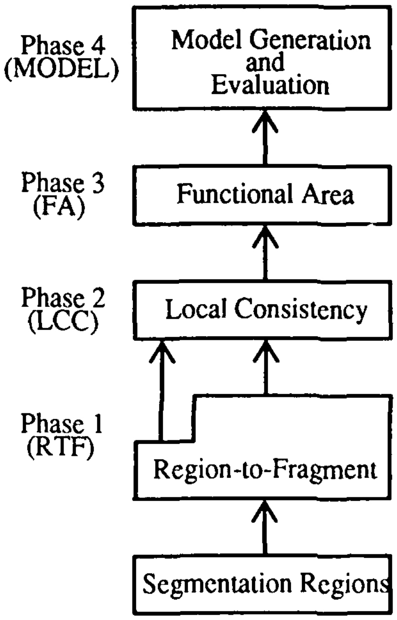


Figure 2: Interpretation phases in SPAM.

As shown in Figure 2, each interpretation phase is executed in the order given. SPAM drives from a local, low-level set of interpretations to a more global, high-level, scene interpretation. There is a set of hard-wired productions for each phase that control the order of rule executions, the forking of processes, and other domain-independent tasks. However, this "bottom-up" organization does not preclude interactions between phases. For example, prediction of a fragment interpretation in *functional-area (FA)* phase will automatically cause SPAM to reenter *local-consistency check (LCC)* phase for that fragment. Other forms of top-down activity include stereo verification to disambiguate conflicting hypotheses in *model-generation (MODEL)* phase and to perform linear alignment in *region-to-fragment (RTF)* phase.

Another way to view the flow of processing in SPAM is that knowledge is used to check for consistency among hypotheses; contexts are created based on collections of consistent hypotheses, and are then used to predict missing components. A collection of hypotheses must combine to create a context from which a prediction can be made. These contexts are refinements or spatial aggregations in the scene. For example, a collection of mutually consistent runways and taxiways might combine to generate a runway functional area. Rules that encode knowledge about runway functional areas may predict that certain sub-areas within that functional area are good candidates for finding grassy areas or tarmac regions. However, an isolated runway or taxiway hypothesis cannot directly make these predictions. In SPAM the context determines the prediction. This serves to decrease the combinatorics of hypothesis generation and to allow the system to focus on those areas with strong support at each level of the interpretation.

SPAM Phase	RTF	LCC	FA	MODEL	Total
Total CPU Time (hours)	1.5	144.5	7.3	0.71	154.01
Total Productions Fired	11274	185950	10447	3085	210756
Effective Productions/Second	2.08	0.357	0.397	1.20	0.380
Total Hypotheses	466	N/A	44	1	N/A

Table 1: San Francisco Airport (log #63)

SPAM Phase	RTF	LCC	FA	MODEL	Total
Total CPU Time (hours)	2.5	17.9	7.3	0.33	28.03
Total Productions Fired	18319	32751	1483	1516	54069
Effective Productions/Second	2.03	0.508	0.056	1.27	0.536
Total Hypotheses	247	N/A	57	1	N/A

Table 2: Washington National Airport (log #405)

SPAM Phase	RTF	LCC	FA	MODEL	Total
Total CPU Time (hours)	0.25	4.12	2.33	0.33	7.03
Total Productions Fired	4713	36949	1503	3774	46939
Effective Productions/Second	5.24	2.30	0.160	3.02	1.85
Total Hypotheses	199	N/A	27	1	N/A

Table 3: NASA Ames Moffett Field (log #415)

Tables 1, 2, and 3 give statistics for run-time and number of production firings for each interpretation phase in SPAM for each of the three airports used in this study: *San Francisco International (SF)* , *Washington National (DC)*, and *NASA Ames Moffett Field (MOFF)* . It is interesting to note that LCC and FA phases account for most of the overall time in a complete run. Further, within these phases much of the RHS evaluation is performed outside OPS5 using external processes. For example, FA spends much of its time doing RHS evaluation outside of OPS5. RTF, on the other hand, spends most of its time within the traditional OPS5 evaluation model and consumes less time than FA, even though it executes a comparable number of productions. It is also clear from these tables that the application of spatial constraints in LCC makes it by far the most expensive phase in terms of amount of time spent, number of productions, as well as number of production firings.

During the LCC phase, knowledge of the structure or layout of the task domain (i.e. airports or suburban housing developments) is used to provide spatial constraints for evaluating consistency among fragment hypotheses. For example, *runways intersect taxiways* and *terminal buildings are adjacent to parking apron* are examples of the kinds of constraints that are applied to the airport scene segmentation. It is important to assemble a large collection of such consistency knowledge since the results of these tests are used to assemble fragment hypotheses found to be mutually consistent as contexts for further interpretation within the functional area phase.

As a result of this preliminary analysis we decided to focus our initial efforts on the parallel implementation of the LCC phase. Another rationale for this approach is the observation that this phase has the largest potential for growth. If a single new scene primitive is added within the RTF phase, many constraints may be added in the LCC phase in order to describe the spatial relationships (and constraints) between each of the other primitives. For these reasons, we believe that as new knowledge is added to the existing SPAM system, the proportion of time can only increase in the LCC phase.

4. Sources of Parallelism in Production Systems

There are two sources of parallelism in production systems: match parallelism (*MP*) and task-level parallelism (*TLP*). In this section we first discuss existing results in match parallelism. We then discuss task-level parallelism and introduce a taxonomy for describing various approaches to achieving effective speed-ups.

4.1. Match Parallelism

In general, production systems spend most of their time (> 90%) in the match phase of the recognize-act cycle. This makes it imperative that we speed up the match as much as possible. In the past few years, an increasing number of researchers have explored many alternative ways to speed up the match in production systems using parallelism [3, 5, 7, 14, 16, 19, 20].

Our own efforts in speeding up the match have culminated in ParaOPS5 [7, 9], a highly optimized C-based parallel implementation of OPS5 for shared memory multi-processors. ParaOPS5 represents our current technology for achieving match parallelism within systems such as SPAM. This implementation parallelizes the highly efficient Rete [4] match algorithm. ParaOPS5 exploits parallelism at a fine granularity: subtasks execute only about 100 instructions. ParaOPS5 has been able to provide significant speed-ups for OPS5 systems that are match-intensive. Figure 3 shows the speed-ups achieved with our current implementation for three different *match intensive* systems: Rubik, Weaver and Tourney. The speed-ups are for an implementation on the Encore Multimax and are reproduced from [7]. Though Rubik and Weaver are seen to achieve good speed-ups, the speed-up in Tourney is quite low. The speed-ups are a function of the characteristics of the productions in the production system (see [6, 7].)

Although systems such as ParaOPS5 have achieved good speed-ups, the total possible speed-up via MP in current production systems is limited (only 20 to 40 fold [5]). This limit is imposed by:

1. *The recognize-act cycle of OPS5*: The OPS5 model requires a synchronization in it's resolve phase. Thus MP is limited to individual cycles; we cannot extract MP across cycles.

2. *Limited match effort per cycle*: In every recognize-act cycle, only a limited number of productions are *affected*, i.e., the match effort per cycle is also quite limited.

Furthermore, MP is based on the assumption that the match phase dominates the entire computation. However, it is possible that the system under consideration is embedded in some other computationally demanding environment. In such cases, it is necessary to parallelize the rest of the computation besides match. Consider a system that spends only 50% of its time in match. Even if the match is made infinitely fast, the total speed-up possible will be only a factor of two (Amdahl's law).

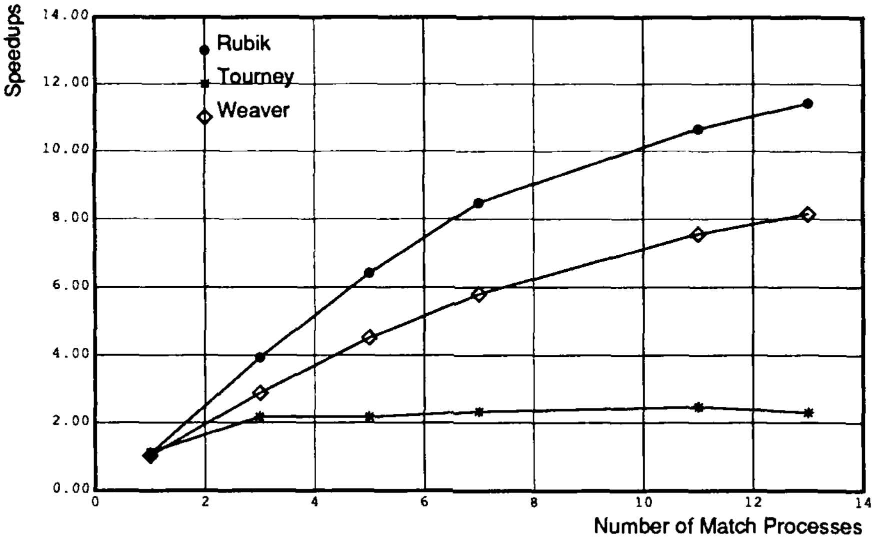


Figure 3: Speed-ups for OPS5 on the Encore Multimax [7].

4.2. Task-Level Parallelism

The limitations of match parallelism described in the previous section encourage the investigation of task-level parallelism. TLP has also been referred to as application parallelism [5], concept parallelism [17], and parallel rule firings [8]. The idea is to use knowledge about the problem domain to create a task decomposition suitable for parallel execution. Our choice of the term TLP for this source of parallelism is partly historical and partly dictated by the inadequacy of the other terms to cover the kind of parallelism provided by production systems like Soar [10].

A system exploiting TLP would be implemented on top of a system exploiting MP. The speed-ups obtained from these two sources can be independent and therefore multiply. We can understand TLP by considering the possible dimensions in which TLP can be divided. These dimensions are:

- *Synchronous/Asynchronous*: Synchronous production-firing systems always require a synchronization in the resolve phase of the recognize-act cycle. All the productions are matched in parallel. In the resolve phase, one or more of the productions are selected for firing. In the act phase, the selected productions are fired in parallel.

  In asynchronous production-firing systems there is no requirement for a synchronization in the resolve phase across processors. Thus, these systems do not have distinct match, resolve and act phases across the parallel system.

  Synchronous systems are less capable of handling variances in processing times for subtasks [15]. As shown in [15], given a fixed amount of work, in the presence of variance, a synchronous system quickly reaches saturation speed-ups, while an asynchronous system can continue to exploit linear speed-ups. So, in a production system embedded in an computationally intensive environment, if executing the RHS of certain productions takes much longer than others, the performance of the synchronous system will degrade heavily. However, synchronous systems may be preferred in the development and debugging stages.

- *Implicit/Explicit*: The parallelism is implicit if the system or the compiler has to *extract* parallelism out of the existing OPS5 code. This requires an analysis of the interference caused by firing productions in parallel. Thus, this is taking a dusty deck view of OPS5 programs.

  Explicit parallelism refers to providing explicit information to the system for exploiting TLP. Thus, the system may be supplied with the information that certain parts of a given task can be solved in parallel, or that certain productions can always be fired in parallel.

  In implicit parallelism, if the system engages in extracting this parallelism at compile-time, then its extraction of parallelism has to be very conservative, as the variable-bindings are unknown. If parallelism is extracted at run-time, then there are overhead costs payed at run-time. These overheads are sequential, and hence can cause considerable slowdowns. A system for exploiting explicit parallelism is able to avoid these problems.

  When the parallelism is implicit, the granularity is usually at the level of productions; it seems difficult to discover a higher level of granularity with implicit parallelism. With explicit parallelism, the user has the freedom to choose the right granularity. The level of granularity is a complex tradeoff of the number of processors available, architectural parameters, variances, data structures and task management overheads. We will discuss the granularity issue in detail in Section 5.

- *Rule distribution/working memory element distribution/No distribution*: This separation is related to the implementation of a parallel rule firing system. In the implementation of a parallel rule-firing system, it is possible to distribute the productions (rules) among processors, where each production set has its own conflict set. This distribution could be done automatically or with the help of the user. However, optimal distribution of productions among processors is a difficult problem.

  A second approach is to allocate all the productions to each processor; the working memory elements are then distributed among the processors. A third approach involves no distribution at all. Here, the parallel rule-firing is built into the control structure of the system.

Table 4 shows the various dimensions and the classification of various parallel rule-firing systems along these dimensions. These dimensions will help to investigate the TLP in SPAM/PSM. The table uses the names of authors to represent systems that do not have any names. Superscripting each system name, we indicate the third dimension that classifies the type of distribution used: rule-distribution, working memory element distribution, or none.

The SPAM/PSM system is the system described in this paper; we will discuss our design choice in detail in Section 5. These dimensions are not intended to be binary; rather, different systems could take different positions along a continuum in these dimensions. However, in the interests of clarity, the table makes a binary division. For instance, the system in [17] is classified as using implicit parallelism — however, it uses some explicit parallelism. It should be noted that except for Soar and SPAM/PSM, all other systems present simulation results on mini-production systems (with 50 or less productions).

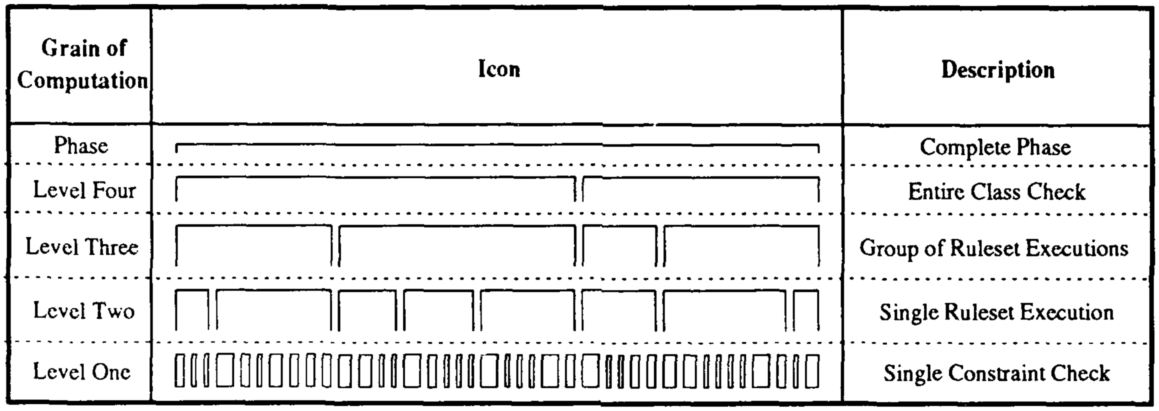
Dimensions	Synchronous	::	Distribution	Asynchronous	::	Distribution
Implicit	Ishida & Stolfo [8]	::	Rule			
	Oshisanwo & Dasiewicz [17]	::	Rule			
Explicit	Soar [5] [10]	::	None	SPAM/PSM	::	WME

Table 4: Dimensions of task-level parallelism.

5. Implementation Methodology

In this section, we develop a methodology for applying task-level parallelism within the context of SPAM. We use knowledge about the task domain to specify several hierarchical task decompositions of the problem in which parallelism can be exploited. Thus, the characteristics of the SPAM task fit the requirements for exploiting task-level parallelism along the explicit dimension described in Section 4.2.

As described in Section 3.2, we will concentrate on the local-consistency phase (LCC) of SPAM for parallelization[3]. The LCC phase applies geometric knowledge (constraints) from the selected domain to the set of interpretations made from the dataset. This application of geometric knowledge can be logically decomposed into several levels, where the tasks within each level are independent and can be performed in parallel. This is illustrated in Figure 4.



Grain of Computation	Icon	Description
Phase		Complete Phase
Level Four		Entire Class Check
Level Three		Group of Ruleset Executions
Level Two		Single Ruleset Execution
Level One		Single Constraint Check

Figure 4: Levels of processing in SPAM LCC.

These levels of decomposition are described below:

- *LCC Phase*: At the highest phase level, the computation is for the entire LCC phase.
- *Level 4*: The phase level computation may be decomposed into tasks at Level 4, where each task applies multiple constraints to a single class of objects. For instance, a task may apply multiple constraints to all objects of class terminal building.
- *Level 3*: A single task at Level 4 may be decomposed into multiple tasks at Level 3. A task at Level 3 applies multiple constraints to a single object within the class of objects selected at Level 4. For example, a Level 3 task may apply multiple constraints to a single terminal building object.
- *Level 2*: A single task at Level 2 involves applying a single constraint to a single object. Thus, a task at Level 2 may apply a constraint such as, access roads lead to terminal buildings, to a single terminal building chosen for a task at Level 3.
- *Level 1*: A single task at Level 2 may have several components to check in applying a constraint to an object. Thus a constraint such as, access roads lead to terminal buildings, requires several roads be checked against the terminal building. A task at Level 1 would perform one of these constraint components.

Within a level, each task involves the firing of from 3 to 100 productions. As mentioned in Section 4.2, an implicit approach to extracting parallelism would make it difficult to obtain parallelism at a higher level of decomposition than individual production firings. Therefore, for this application, an explicit approach to parallelism is more appropriate.

With an explicit approach to parallelism, the choice of the right level of decomposition, or the right granularity, for parallelization must be made. This choice is determined by several factors:

1. *Task granularity*: As the average time per task gets smaller, task management overheads

[3] Since the analysis is performed using the original, expensive Lisp-based SPAM system, we have extracted a representative subset of the three airport datasets to drive the analysis.

will have a greater impact and communication overheads and system resource contention will become more of a bottleneck.

2. *Ratio of tasks to processors*: The achievable parallelism is bounded by the number of available processors. At lower task to processor ratios, a large variance in task processing time will have a negative impact on processor utilization and the speed-ups obtained from parallelism. With higher ratios, the impact is less pronounced.
3. *Coefficient of variance*: Defined as σ/μ, this provides a means of normalizing, for different levels of decomposition, the effect of variance in task granularity on processor utilization. A high coefficient of variance will reduce processor utilization, resulting in lower speed-ups. This effect is more severe in synchronous systems.
4. *Decomposition effort*: This is a somewhat subjective measure. Proceeding down the hierarchy of levels, each task at the current level must be decomposed into several tasks at the next level of granularity. Usually, more work is required to specify the decomposition and design an implementation at the lower levels. The benefits of the additional parallelism that can be achieved at a lower level relative to the effort required must be assessed.

In order to choose the right level of decomposition at which to parallelize the SPAM LCC phase, we instrumented the SPAM system to obtain measurements at each level for the number of tasks and their run-time average, standard deviation, and coefficient of variance. The results of these measurements for each of the three airport datasets is presented in Tables 5, 6, and 7.

Using information from Tables 5, 6, and 7, the appropriate level of granularity can now be chosen. For Level 4, the task to processor ratio is smaller than one, so we immediately rejected pursuing parallelism at this level. Levels 3 and 2 are very similar to each other in that they have enough tasks, their variances are not large, and the task granularities are much larger than the expected task management and communication overheads. Both levels, therefore, seemed to us to be worthwhile candidates. Level 3 seemed somewhat more desirable as less effort appeared to be required of us to achieve amounts of parallelism similar to that available in Level 2.

Level 1 was rejected for several reasons. First and most importantly, the additional effort involved in decomposing the system at the granularity of Level 1 would not allow us to achieve any more parallelism than at Level 2 or 3 because of the limitation on the number of processors. Second, the task granularity is much smaller and thus closer to the overheads for task management and communication than any of the other levels. Finally, the task to processor ratio is on the order of 1000. This can have a detrimental affect due to the initialization overhead. Our conclusion, then, was to exploit parallelism at the granularity of Levels 2 or 3.

The decomposition methodology can be summarized as follows:

- Analyze the baseline system and determine where the time is going.
- Determine if the explicit dimension of TLP (Section 4.2) is appropriate.
- Characterize the computation in terms of independent task decompositions at different granularities.
- Obtain measurements of the system characteristics for each level of decomposition.
- Analyze the measurements to select a level of decomposition for parallelization.

The second dimension of task-level parallelism addresses the issue of synchronous versus asynchronous execution. With an explicit decomposition at Level 3, there is no synchronization requirement. Furthermore, asynchronous models help in reducing the impact of variance. We therefore decided to decompose the system so as to allow the asynchronous rule-firings.

The final dimension of task-level parallelism addresses the issue of production versus working-memory partitioning. We decided to use working-memory partitioning, as this facilitates the explicit decomposition at the higher granularity.

Level	Average time per task (sec)	Standard deviation (sec)	Coefficient of variance	Number of tasks
Level 4	875.27	525.92	0.601	9
Level 3	65.65	29.51	0.449	120
Level 2	20.90	8.48	0.406	377
Level 1	0.489	0.0782	0.159	16104

Table 5: Average, standard deviation and coefficient of variance for SF.

Level	Average time per task (sec)	Standard deviation (sec)	Coefficient of variance	Number of tasks
Level 4	1308.66	641.72	0.490	9
Level 3	78.51	30.48	0.388	150
Level 2	24.04	9.51	0.396	490
Level 1	0.430	0.0677	0.157	27399

Table 6: Average, standard deviation and coefficient of variance for DC.

Level	Average time per task (sec)	Standard deviation (sec)	Coefficient of variance	Number of tasks
Level 4	165.60	121.20	0.732	9
Level 3	20.07	8.02	0.399	74
Level 2	5.57	2.43	0.436	268
Level 1	0.349	0.0455	0.130	4274

Table 7: Average, standard deviation and coefficient of variance for MOFF.

6. SPAM/PSM Implementation

This section describes the SPAM/PSM system that implements the LCC phase of SPAM described in Sections 3.2 and 5. The system is built on top of the ParaOPS5 system described in Section 4.1. The SPAM/PSM system is implemented on an 16-processor Encore Multimax, a shared-memory multiprocessor based on the National Semiconductor NS32332 processor, rated at approximately 1.5 MIPS.

6.1. SPAM/PSM Architecture

Figure 5 gives a process hierarchy view of the SPAM/PSM system for the LCC phase. Viewed from the top level, the execution model consists of a *control process*, a set of *task processes*, and a *queue of tasks* to be executed. The size and number of tasks in the queue reflects the level of decomposition chosen for the LCC phase. The decomposition of LCC was described in Section 5.

The control process takes the output from the phase preceding SPAM's LCC phase and builds the queue of tasks. It then forks the task processes and, once they have completed all the tasks, collects from them the results that will be passed on to the next SPAM processing phase.

Each of the task processes is a complete and independent ParaOPS5 system. Thus, each task process

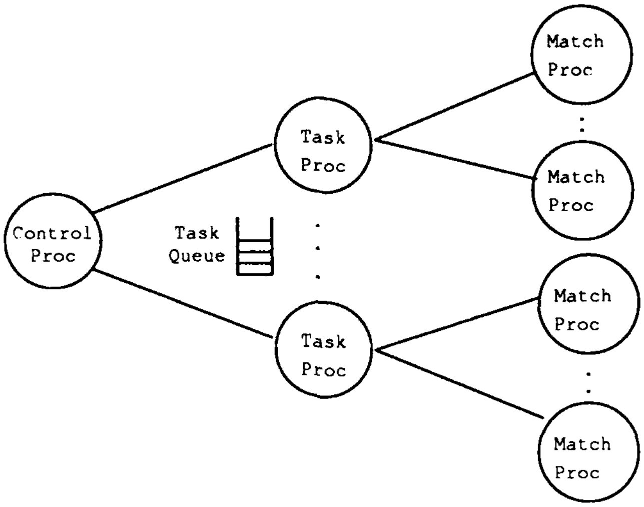


Figure 5: Organization of the SPAM/PSM system.

has its own working memory, conflict set, Rete node memories, etc. Each task process has a production memory, wh'ch represents all the productions in the system, and effectively has a copy of the initial working memory supplied by the control process. At system initialization time, each task process can also fork a set of match processes (see Figure 5) which will perform the match in parallel.

The work performed by the SPAM/PSM system to carry out the LCC phase involves a task process removing a task from the queue and executing its ParaOPS5 system on that task. The task itself is just a WME which, when added to the process' Rete network, initializes the production system. Thus, each task can be characterized as the execution of an independent OPS5 program.

In the absence of the match processes, a task process performs the usual ParaOPS5 role of match, conflict resolution, and production firing, to carry out the OPS5 recognize-act cycle. If dedicated match processes are present, they perform the match instead, providing a second and independent axis of parallelism in the SPAM/PSM system. When there are no productions left to fire, the task is complete, and the task process goes to the queue for another task.

Thus, the SPAM/PSM system realizes our specifications:

1. *Explicit parallelism:* The decomposition of the LCC phase is explicitly specified. The task queue is initialized with independent tasks, depending on the level of decomposition, in the beginning of the run.

2. *Asynchronous production firing:* All the task processes are independent ParaOPS5 systems. Therefore, these processes can fire productions without synchronizing with each other.

3. *Working-memory element distribution:* Each task process has a copy of the entire set of productions. The working memory is distributed among the various task processes.

6.2. Measurement Techniques

The SPAM/PSM system is instrumented to measure the time spent in executing the tasks from two of the LCC phase decompositions, Level 2 and Level 3, identified in Section 5. The control process previously described is used to monitor and time this processing. Measurement begins at the point after which the control process has built the task queue and forked the task processes, and all the task processes have performed their initializations. Speed-ups are computed by comparing the measured execution time against the execution time of the BASELINE version, which consists of the control process, one task process, and no dedicated match processes.

Because of the 16 processor limit, we measure the effects of task-level parallelism and match parallelism in isolation. We allocate one processor for the control process, which is used only to time and not to perform tasks, and we allow one processor to the operating system. This permits us to vary the number of task processes from 1 to 14 in the isolated measurement of task-level parallelism. Next we measure the effect of match parallelism in isolation by using a single task process and varying the number of dedicated match processes from 0 to 13.

We are then able to use these two separate measures of task-level parallelism and match parallelism to predict the combined effect of the two. However, with 14 available processors, we are able to test only a subset of the possible combinations. For example, 4 task processes, each having 2 dedicated match processes, uses 12 processors (4 + (4 * 2)). Thus, dedicating 3 match processes requires 16 processors (4 + (4 * 3)) and, therefore, cannot be accommodated.

7. Results and Analysis

In this section we present the results of our parallel implementation, SPAM/PSM, of the SPAM LCC phase run on these three different airport datasets: SF, DC, and MOFF. As described above, the speed-ups are obtained for applying task-level parallelism and match parallelism in isolation and then for a combination of the two. We obtained results for two of the parallel decompositions, Level 3 and Level 2, identified in Section 5.

It is important to note that all the speed-ups are computed against a baseline system which represents an optimized uniprocessor implementation of the SPAM LCC phase. The original SPAM system is implemented in Lisp, using an unoptimized Lisp-based OPS5. It forks independent processes to perform geometric computations in the RHS. We ported the LCC phase of the system to C and ParaOPS5 and replaced the forked computational processes with C function calls. This baseline system itself provides approximately a 10-20 fold speed-up over the original Lisp-based implementation for the LCC phase on the three datasets used here.

7.1. The Baseline System

The baseline version of the system uses a single task process to execute all the tasks in the system. The results from this version are given in Table 8 and provide a picture of the magnitude of the LCC phase. The column marked DATASET gives the name of the airport and the decomposition level used. The column marked TOTAL TIME shows the total time to execute all the tasks from the queue for the given number of tasks executed. The average time per task is then shown in the next column. Finally, we further characterize the LCC phase with the total number of productions fired (PRODS FIRED), RHS actions performed (RHS ACTIONS), and changes to working memory (CHANGES TO WM).

Dataset	Total time (sec)	Number of tasks	Average time per task (sec)	Prods fired	RHS actions	Changes to WM
SF Level 3	1433	283	5.07	33475	42383	39116
SF Level 2	1423	941	1.51	32251	41159	38550
DC Level 3	988	151	6.55	20059	31205	26714
DC Level 2	956	490	1.95	19418	30564	26412
MOFF Level 3	991	209	4.74	22203	23637	23368
MOFF Level 2	973	700	1.39	21294	22728	22950

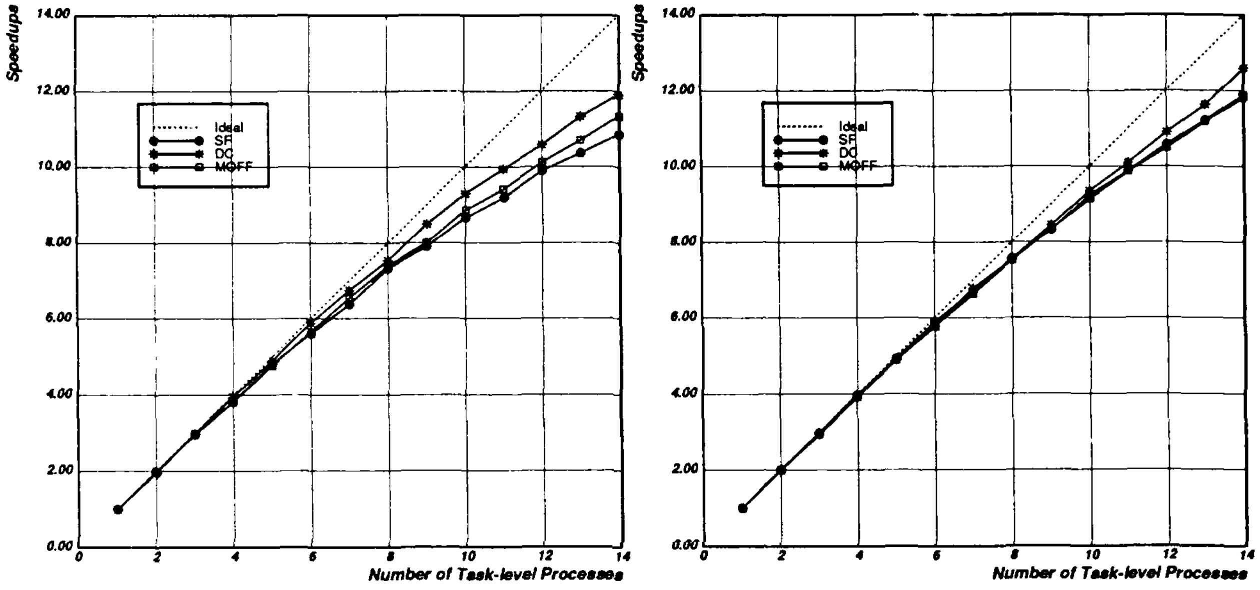
Table 8: Measurements for baseline system on the datasets[4].
(Represents the optimized, ParaOPS5-based, uniprocessor version.)

[4] These datasets are larger than those shown in Tables 5, 6, and 7.

The execution times in Table 8 provide the basis for computing all of the speed-ups. For a given airport dataset, there is a small difference in the total execution time between the two levels of decomposition. These differences arise due to the diffbrences in the initial set of productions fired for generating the *tasks* for the two levels.

7.2. Speed-ups due to Task-Level Parallelism

The results of applying task-level parallelism are shown in Figure 6. The speed-up curves shcw near linear speed-ups for both levels of decomposition. The speed-ups within a level are almost the same among the three airport datasets. The maximum speed-up achieved using 14 processors is 11.90 fold in Level 3 and is 12.58 fold in Level 2.



Speedup at Level 3

Speedup at Level 2

Figure 6: Speed-ups varying the number of task-level processes.

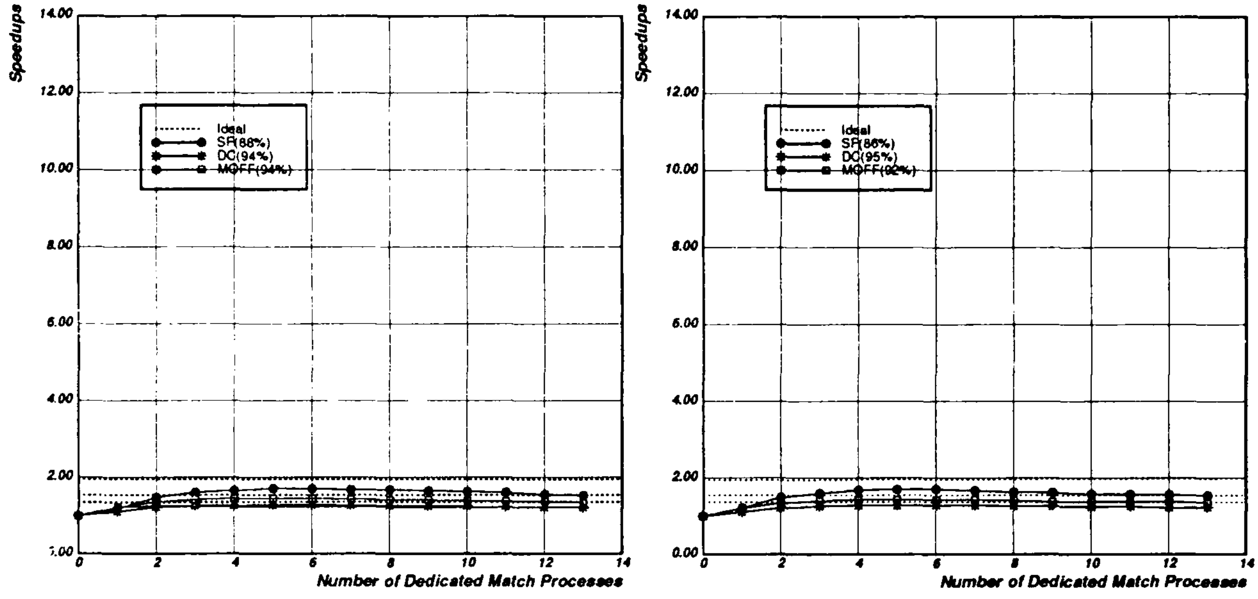
Across the two levels, we see that the curves are consistently better in Level 2, although by only a small factor (less than 10%). While the difference is small, Level 3, with its higher granularity, was expected to have the edge in speed-up, since its task management overheads would be lower. However, the task management overheads in both levels are very low: less than .25 seconds, or less than .1% of the processing time for all the tasks. Moreover, the coefficient of variance for tasks at both levels was seen to be the same in Section 5.

Further investigation of the individual processing times of the tasks in the queue showed that there are a few tasks in each level that have execution times that are an order of magnitude larger than the average task in that level. Some of these tasks occur at the end of the task queue and create a tail-end effect in which processor utilization is low at the end of the phase. The relative disparity of these large tasks is greater within Level 3 and thus accounts for the slightly better speed-ups in Level 2.

One way to both negate this disparity and reduce the tail-end effect would be to use a separate task queue for the larger tasks and process them at the beginning of the phase. This would result in better processor utilization and thus better speed-up curves in both levels. SPAM can provide the necessary information to indentify the sizes of the tasks. This and other related issues of scheduling tasks are subjects for future work.

7.3. Speed-ups Due to Match Parallelism

Figure 7 shows the results for applying match parallelism to each of the tasks in a parallel decomposition for Levels 2 and 3. Match parallelism is obtained by dedicating processes to perform the match within the OPS5 recognize-act cycle. Since the baseline version of the system has only a task



Speedup at Level 3: Asymptotic Limits SF=1.95 DC=1.36 MOFF=1.54

Speedup at Level 2: Asymptotic Limits SF=1.99 DC=1.36 MOFF=1.55

Figure 7: Speed-ups varying the number of match processes.

process and no dedicated match processes, it is represented in both graphs at position 0 on the horizontal axis. From the graphs, we see that applying match parallelism to the LCC phase yields very different speed-up results from those achieved using task-level parallelism. As stated in Section 4, the theoretical maximum speed-up that can be obtained is limited according to the percentage of total execution time spent in match.

The dotted lines on the graphs show the theoretical speed-up limits. For Level 3, these limits are 1.95, 1.36, and 1.54 for SF, DC, and MOFF respectively. We were able to obtain respective speed-ups of 1.71, 1.28, and 1.45 which represent 88%, 94%, and 94% of the corresponding asymptotic limits. In all three cases, the speed-ups peaked using 6 or less match processes. Similar results are shown for Level 2.

7.4. Multiplicative Speed-ups

To validate the multiplicative effect of the two independent axes of parallelism [18], the system was run using task-level and match parallelism in consort. While the scope of the experiments was limited by the small number of processors, the speed-ups obtained in these combined runs were consistent with the speed-ups predicted by the multiplication of speed-ups from the two separate sources. Table 9 shows the results of some of these combined runs on SF for Level 2. The top row of the table varies the number of dedicated match processes from 0 to 5. The left column of the table varies the number of task processes from 1 to 7. The first row of numbers in the table gives the speed-ups from match parallelism in isolation. The first column of numbers in the table gives the speed-ups from task-level parallelism in isolation.

The table entry at ($Task_1$, $Match_0$) represents the baseline version of the system. Each of the other table entries shows the achieved multiplicative speed-up from the combined sources with the predicted speed-up in parentheses directly below. For example, the entry ($Task_4$, $Match_2$) represents the use of 4 task processes with each having 2 dedicated match processes. The achieved speed-up for this configuration is 5.82 fold and the predicted speed-up is 5.96 (3.98*1.50). Table entries marked with an asterisk could not be measured due to a lack of processors on the machine (see Section 6.2). For example, ($Task_4$, $Match_3$) requires 17 processors: 1 control process, 4 task-level processes, and 12 (= 4*3) dedicated match processes. The table shows the achieved speed-ups to be very close to the predicted speed-ups. Similar results were obtained for DC and MOFF.

The speed-up curves for task-level and match-level parallelism graphically indicate that the benefits from task-level parallelism are much more significant than from match parallelism. Thus, in a setting

	Match$_0$	Match$_1$	Match$_2$	Match$_3$	Match$_4$	Match$_5$
Task$_1$	1	1.21	1.50	1.60	1.68	1.70
Task$_2$	2.01	2.42 (2.43)	2.97 (3.01)	3.16 (3.22)	3.30 (3.37)	3.36 (3.42)
Task$_3$	2.98	3.57 (3.60)	4.42 (4.46)	4.73 (4.78)	* (5.01)	* (5.07)
Task$_4$	3.98	4.73 (4.81)	5.82 (5.96)	* (6.37)	* (6.69)	* (6.77)
Task$_5$	4.93	5.82 (5.95)	* (7.39)	* (7.89)	* (8.28)	* (8.38)
Task$_6$	5.89	6.98 (7.12)	* (8.83)	* (9.42)	* (9.90)	* (10.01)
Task$_7$	6.70	8.04 (8.09)	* (10.05)	* (10.72)	* (11.26)	* (11.39)

Table 9: Multiplicative speed-ups in SPAM/PSM for SF Level 2. Parenthesized numbers are the predicted speedups.

where the number of available processors is limited, it is best to allocate them to task-level parallelism rather that match parallelism. We believe that the potential for additional speed-ups in SPAM from task-level parallelism is quite high; an expectation of 50 to 100 fold does not seem unreasonable, since:

1. The tasks within any of the LCC decompositions are independent of one another.
2. Several hundred tasks are available in Level 2.
3. The task queue management overheads measured for Level 2 and Level 3 are very low, especially with respect to the task granularity, and thus are not a factor.

The current scheme of decomposition depends on a centralized task-queue for effective distribution of tasks among processes. A centralized task queue may become a bottleneck for an increasing number of processes; therefore, we need to investigate schemes for effective distribution of tasks among processes.

Though our scheme of parallelization has been presented in the context of non-match-intensive system, the scheme is applicable to match-intensive systems as well. In match-intensive systems, match parallelism will make a substantial contribution to the speed-ups.

8. Summary and Conclusions

In this paper we characterized task-level parallelism in production systems along three dimensions and, from that, selected an explicit, data-driven, asynchronous approach for exploiting it. The system we presented, SPAM/PSM, is a real, computationally demanding, high-level vision system that relies on knowledge-based reasoning. With the SPAM/PSM system, we showed that an explicit approach to task-level parallelism can yield significant speed-ups.

The explicit approach relies on knowledge that the system designer has available about the nature of the problem. The designer uses this knowledge directly to arrive at a problem decomposition in which parallelism can be exploited. The decomposition is made based on the data upon which the system must operate and several levels of decomposition are possible. We saw that the choice of the correct level at which to exploit parallelism is based upon a number of factors; among these are the task granularity, task management and communication overheads, the variance in task processing times, and the ratio of total tasks to processors.

For the SPAM/PSM system we presented a methodology for obtaining a parallel task decomposition and arrived at three levels of decomposition. We implemented two of those levels and obtained near linear speed-ups with a maximum of over 12 fold using 14 processors. The results obtained indicate that speed-ups on the order of 50 to 100 fold from task level parallelism might be realized on a machine with a comparably large number of processors. We believe that the success achieved with the SPAM/PSM system gives hope to designers of other rule-based systems to realize systems with much lower execution times by applying task-level parallelism. Also the potential for very large speed-ups indicated here should serve as encouragement to the designers of large-scale multiprocessor systems.

We also obtained results for applying match parallelism to each of the tasks in a parallel decomposition. We saw that this speed-up represented an independent axis of parallelism and thus could be multiplied with the speed-up obtained from task-level parallelism. In the airport data sets tested, this axis provided a factor of 1.5 to 2 fold parallelism.

We believe that the explicit, data-driven approach taken here holds better potential for realizing task-level parallelism than implicit approaches that attempt to extract parallel rule-firings using a task-independent, bottom-up analysis. With these latter kinds of approaches to task-level parellelism, there is not enough information available at compile-time to make these decisions and the complexity and overhead required at run-time to perform the analysis is prohibitive and does not seem likely to yield much speed-up from parallelism. Such parallel rule-firing schemes are still constrained by the overall synchronous nature of the OPS5 recognize-act cycle. In addition, the run-time analysis for parallel rule-firings places another synchronous constraint upon the system which presents a further bottleneck to parallelism. The top-down, explicit approach presented here achieves parallel rule-firings without this synchronous constraint and the overhead of the run-time analysis. Furthermore, the analysis required to arrive at a suitable parallel decomposition is straightforward and can be arrived at fairly quickly.

Finally, the framework for exploiting task-level parallelism presented in this paper seems most suitable for parallelizing knowledge-intensive systems that exhibit weak interaction between the individual subtasks of the task. This framework is especially useful for systems with a large computational demand separate from the demand imposed by match.

9. Future Work

In the near future we plan to make our SPAM/PSM implementation a useful tool for SPAM researchers. This means partitioning and parallelization of the other phases of SPAM besides the local consistency check phase. Moreover, currently, the initialization subphase within the local-consistency phase consumes a large amount of processing time. We need to optimize and/or parallelize the initialization subphase.

Results from Section 7 show that large amounts of parallelism can be exploited in SPAM, and thus, significantly larger number of processors could be employed in exploiting the parallelism. Shared-bus multi-processors like the Encore Multimax cannot support such a large number of processors. We need to evaluate other scalable parallel architectures for exploiting match and task-level parallelism in production systems. Toward this end, we are currently investigating implementations on message-passing computers; simulation results for production systems on message-passing computers [1] have shown positive results.

Our long term plan is the investigation of task-level parallelism in systems besides SPAM. We hope such investigations will help us refine the general methodology for exploiting task-level parallelism in production systems.

10. Acknowledgements

We'd like to thank the members of both the PSM group and Digital Mapping Lab for helpful discussions. In particular, discussions with Anurag Acharya, Lanny Forgy, and Brian Milnes did much to shape the design of the current system. Matt Diamond provided invaluable programming assistance, and Kathy Swedlow helped make our prose a bit more understandable.

References

1. Acharya, A. and Tambe, M. Production Systems on Message Passing Computers: Simulation results and analysis. Computer Science Department, Carnegie Mellon University, December, 1988
In preparation.

2. Brownston, L., Farrell, R., Kant, E., and Martin, N.. *Programming Expert Systems in OPS5: An introduction to rule-based programming*. Addison-Wesley, Reading, Massachusetts, 1985.

3. Butler, P. L., Allen, J. D., and Bouldin, D. W. Parallel architecture for OPS5. Proceedings of the Fifteenth International Symposium on Computer Architecture, 1988, pp. 452-457.

4. Forgy, C. L. OPS5 User's Manual. Tech. Rept. CMU-CS-81-135, Computer Science Department, Carnegie Mellon University, July, 1981.

5. Gupta, A. *Parallelism in Production Systems*. Ph.D. Th., Computer Science Department, Carnegie Mellon University, March 1986. Also available in *Parallelism in Production Systems*, Morgan-Kaufman, 1987.

6. Gupta, A., Forgy, C. L., Kalp, D., Newell, A., and Tambe, M. Parallel OPS5 on the Encore Multimax. Proceedings of the International Conference on Parallel Processing, August, 1988, pp. 271-280.

7. Gupta, A., Tambe, M., Kalp, D., Forgy, C. L., and Newell, A. "Parallel implementation of OPS5 on the Encore Multiprocessor: Results and analysis". *International Journal of Parallel Programming 17*, 2 (1988). In press.

8. Ishida, T. and Stolfo, S. Towards the parallel execution of rules in production system programs. Proceedings of the International Conference on Parallel Programming, August, 1988, pp. 568-574.

9. Kalp, D., Tambe, M., Gupta, A., Forgy, C., Newell, A., Acharya, A., Milnes, B., and Swedlow, K. Parallel OPS5 User's Manual. Tech. Rept. CMU-CS-88-187, Computer Science Department, Carnegie Mellon University, November, 1988.

10. Laird, J.E., Newell, A. and Rosenbloom, P.S. "Soar: An architecture for general intelligence". *Artificial Intelligence 33*, 1 (1987), 1-64.

11. McKeown, D.M., Harvey, W.A., and McDermott, J. "Rule based interpretation of aerial imagery". *IEEE Transactions on Pattern Analysis and Machine Intelligence 7*, 5 (1985), 570-585.

12. McKeown, D., McVay, W., and Lucas, B. "Stereo verification in aerial image analysis". *Optical Engineering 25*, 3 (1986), 333-346.

13. McKeown, D., Harvey, W., and Wixson, L. Automating knowledge acquisition for aerial image interpretation. To appear in *Computer Vision, Graphics and Image Processing*.

14. Miranker, D. P. Treat: A better match algorithm for AI production systems. Proceedings of the National Conference on Artificial Intelligence, August, 1987, pp. 42-47.

15. Mohan, J. *Performance of Parallel Programs: Model and Analyses*. Ph.D. Th., Computer Science Department, Carnegie Mellon University, July 1984.

16. Oflazer, K. *Partitioning in Parallel Processing of Production Systems*. Ph.D. Th., Computer Science Department, Carnegie Mellon University, March 1987.

17. Oshisanwo, A. and Dasiewicz, P. A parellel model and architecture for production systems. Proceedings of the International Conference on Parallel Processing, August, 1987, pp. 147-153.

18. Reddy, R. and Newell, A. Multiplicative speedup of systems. In Jones, A., Ed., *Perspectives on Computer Science*, Academic Press, New York, New York, 1977, pp. 183-198.

19. Schreiner, F. and Zimmerman, G. Pesa-1: A parallel architecture for production systems. Proceedings of the International Conference on Parallel Processing, August, 1987, pp. 166-169.

20. Tambe, M., Kalp, D., Gupta, A., Forgy, C.L., Milnes, B.G., and Newell, A. Soar/PSM-E: Investigating Match Parallelism in a Learning Production System. Proceedings of the ICM/SIGPLAN Symposium on Parallel Programming: Experience with Applications, Languages, and Systems, July, 1988, pp. 146-160.

MULTI-SCALE CONTOUR MATCHING IN A MOTION SEQUENCE

Salit Levy Gazit and Gérard Medioni
Institute for Robotics and Intelligent Systems
University of Southern California
Los Angeles, CA 90089-0273

ABSTRACT

This paper presents a method to establish correspondences between elements of two images in a motion sequence, even when this motion induces large displacements in the image.

The matching proceeds by trying to find the most similar shape corresponding to each contour (local process), and by trying to find similar displacements for adjacent contours (global process).

The primitives we use are connected edgels obtained from a process called Adaptive Smoothing, in which an image is repeatedly convolved with a small averaging mask whose coefficients are updated at each iteration to reflect the amount of discontinuity in the local window. This process depends on a single scale parameter, analogous to the space constant of a Gaussian filter. The main advantage of this adaptive process is that the resulting edges have a position independent of the scale, making the tracking of edges across scales a trivial problem. This property allows us to perform the matching of motion images hierarchically in a coarse to fine manner.

We applied the method to several sequences, processing each with parameter values of 16, 8 and 4. The images match well even when the displacements are quite large.

1 INTRODUCTION

Several approaches have been tried for computational analysis of motion from image sequences, and many of them need a set of matching points or matching features for the motion analysis. Therefore matching features between consecutive frames is an important step in motion analysis.

The choice of the appropriate features is crucial. Low level features, such as edgels, are easy to detect, but convey very little context information and therefore are difficult to match correctly. Very high level features, such as surface patches or objects, are easy to match but very difficult to detect. Researchres in the field have used simple features such as edgels [15, 12]; intermediate level features such as line segments [16, 3], vertices [1, 2], local statistics [13, 21], extrema of local grey level curvatures [7], corners [6], regions [17] and high curvature points in zero crossings contours [20]; or complex features such as structured forms [14] or even the images of recognized objects. Coarse-to-fine resolution matching [10, 11] can be used to reduce the complexity of the matching.

Ullman [22] conjectures that in humans correspondence is based neither on luminance alone nor relating complex form, but is rather an intermediate level process based on relatively simple properties of the image.

Line segments embody continuity information, yet are local enough, so the chance that a line segment belongs to two different objects is very small. However, curvature and corner information, very important for matching, are lost. Corners suffer from the opposite problem - they contain curvature information but no continuity information. Also detecting corners is difficult and error prone.

We prefer to use edgel contours of varying lengths as our basic feature. They are easy to detect and provide a meaningful representation for the image. However, matching them presents some problems. It is possible for contours to merge or split, due to occlusion, noise, errors of the linker, etc. Thus, two contours may only partially match, and there is no simple way to detect, in advance, where along the contours this partial match occurs. We try to solve this problem by dividing each contour into a few sections and matching each of them separately. Each of these matches is then increased by adding adjacent points until the similarity starts to decrease, then use a global preference criteria based on the common motion of matching sections as well as shape similarity. Our method relies heavily on length and invariance of shape of the contour sections. Straight or short contours are not matched as well by the algorithm.

To obtain the features, we first smooth the images with adaptive filters [18] at different scales and then apply Canny's edge detector to the smoothed images. The result is a hierarchical set of edgels (but not a hierarchical set of contours, because linking may be performed differently around junctions). Adaptive smoothing does not shift the edgels by more than a pixel between scales and the edgels are accurately positioned. Using a hierarchical set of features solves the problems associated to having to match closely spaced and short contours, which are bound to arise when we use finer scales.

We first review the fundamental properties of Adaptive smoothing filters, then describe our matching algorithm in detail, provide some illustrative examples on real images and describe possible extensions and future research.

2 ADAPTIVE SMOOTHING

The extraction of features such as intensity discontinuities from an image is an essential task in early vision. Those discontinuities usually correspond to the physical boundaries of the objects present in the scene but because of the complexity of the physical world and of the imaging apparatus, and of multiple sources of noise, the image to be processed is complex, and the detection of such discontinuities is non trivial. Furthermore, a crucial idea based on physiological observations is that an image can be interpreted at a few different scales depending on how much details are taken into account. Hence, many researchers in the vision community recently focused on multiple scale features extraction.

Out of the major approaches proposed in the litterature to tackle this challenging problem, Witkin [23] introduced the concept of scale-space: the idea is to embed the original image in a family of derived images $I(x, y, \sigma)$ obtained b. convolving the original image with a Gaussian kernel using different values of σ. Larger values of σ, the scale-space parameter, correspond to images viewed at coarser resolutions. The major drawback of Gaussian smoothing is that features extracted at larger values of σ are usually not localized correctly since edges start interacting. It is then necessary to track coarser features across the different scales downwards to the finest scale in order to find their correct location, which poses a difficult correspondence problem in 2-D.

The purpose of Adaptive Smoothing recently introduced by Saint-Marc and Medioni [18] is to smooth an intensity image (for instance) while preserving intensity discontinuities. This is achieved by repeatedly convolving the image with a very small averaging filter modulated by a measure of intensity discontinuity at each point. A relatively small number of iterations is needed to obtain a smooth image and a parameter k equivalent to σ in Gaussian smoothing fixes the amplitude of the discontinuities to be preserved. Since the latter are preserved, they are directly localized, hence no tracking is needed as opposed to Gaussian smoothing. The adaptive smoothing of an image $I(x, y, n)$ is performed as follows:

$I(x, y, n + 1) = \frac{1}{R} \sum_{i=-1}^{+1} \sum_{j=-1}^{+1} I(x + i, y + j, n)c(x + i, y + j, n)$ with $R = \sum_{i=-1}^{+1} \sum_{j=-1}^{+1} c(x + i, y + j, n)$
where $c(x, y) = f(d(x, y)) = e^{-\frac{|d(x,y)|^2}{2k^2}}$ with $d(x, y) = \sqrt{G_x^2 + G_y^2}$

$d(x, y)$ represents the magnitude of the gradient $(\frac{\partial I}{\partial x}, \frac{\partial I}{\partial y})^\top = (G_x, G_y)^\top$, computed in a 3×3 window.

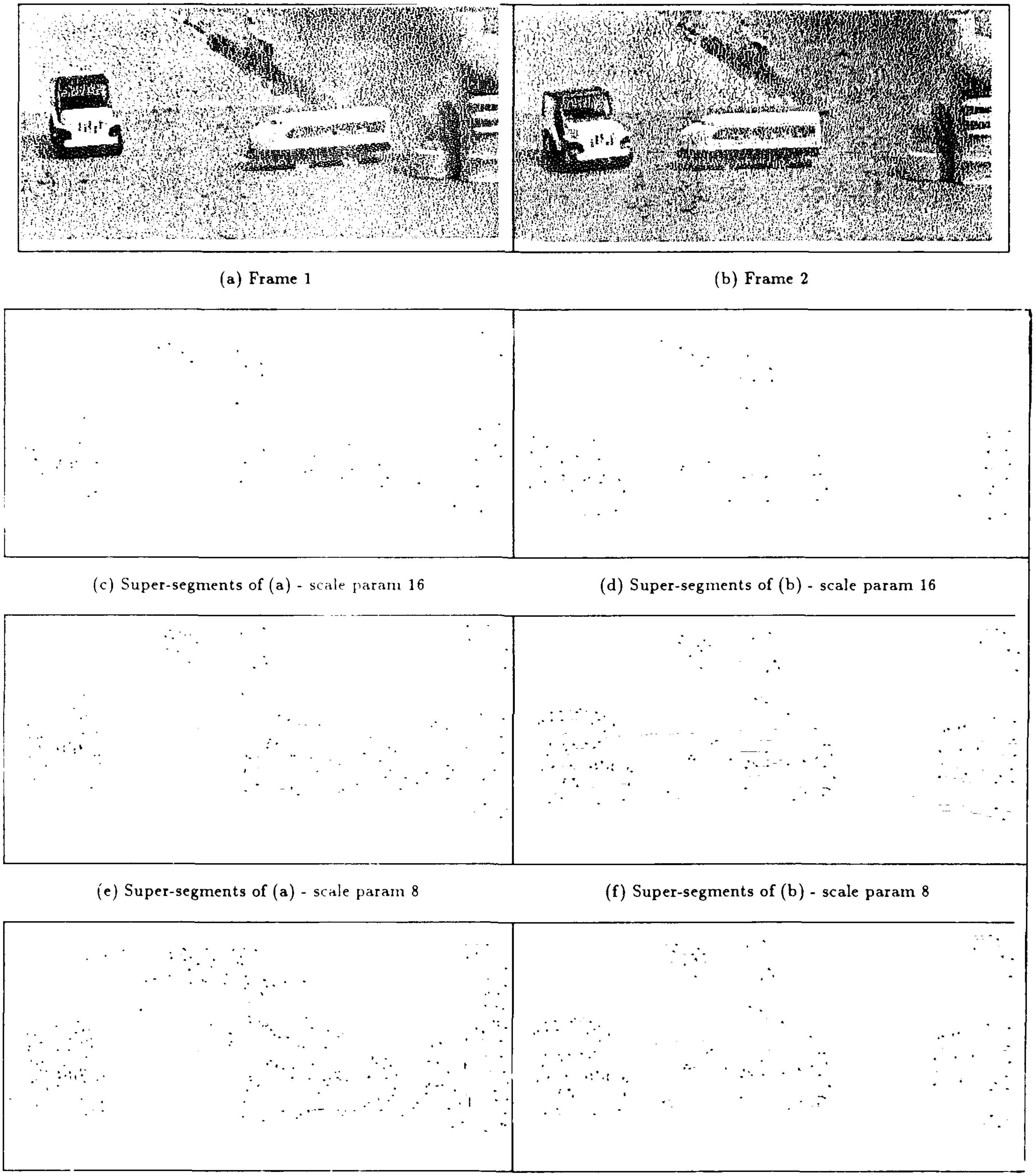
Figure 1 shows the results of edge detection after adaptive smoothing with 3 different values of the parameter k, 16, 8 and 4, for two images of a sequence. In all cases, the number of iterations n was fixed to 10. The results show that the features detected at different scales are very well localized and do not move.

3 THE MATCHING ALGORITHM

The matching method is similar to our previously reported work [9], but we now use multiple scales and better primitives.

3.1 PRIMITIVES

We believe that using sections of edgel contours of varying lengths alleviate many problems arising with previously used features. Complex features, such as surfaces or objects are easy to match, but extracting them correctly is a difficult error-prone task. Lower level features are easier to detect but difficult to match. We believe that intermediate level features are the answer and have previously discussed the merits and problems of two of these, namely line segments and corners. Our primitives are edge contours that combine many merits of both line segments and corners. In our previous work [9] we used contours of zero crossings of (la.ge) Laplacian of Gaussian masks [5]. We got very



(a) Frame 1

(b) Frame 2

(c) Super-segments of (a) - scale param 16

(d) Super-segments of (b) - scale param 16

(e) Super-segments of (a) - scale param 8

(f) Super-segments of (b) - scale param 8

(g) Super-segments of (a) - scale param 4

(h) Super-segments of (b) - scale param 4

Figure 1: Jeep and train image - features found with different scale params

good matches using these features, but the results were hard to use because Gaussian smoothing tends to shift the edgels from their correct positions and may create spurious edgels. To overcome these difficulties, we now use adaptive filters [18] at different scales to smooth the images. We then detect edgels in the smoothed images using Canny's edge detector [4], link them, using an extension of our zero crossings linking algorithm [8] and extract segments and super-segments at each scale.

The advantage of this approach is that smoothing with adaptive filters preserves the causal properties of Gaussian filters without shifting the edgels or creating spurious ones.

We define a *super-segment* as an object that describes an edgel contour. It consists of the directed list of edgels representing the contour, as well as the line-segments approximating it. Each line-segment "knows" the location in the edge list of its beginning and end points. Often we use the term *super-segment* to refer to its edgel contour.

A *section* of a super-segment is a portion of its edgel list of any size.

3.2 HIERARCHICAL MATCHING

The basic matching algorithm is an extension of the algorithm described in [9]. The hierarchical matching algorithm is very simple:

Assume that the images are $image_1$ and $image_2$, the coarser scale is h and the finer one is l.

Then we have 4 sets of super-segments: $S_{1,h}$, $S_{2,h}$, $S_{1,l}$ and $S_{2,l}$, where $S_{i,m}$ is the set of super-segments detected in $image_i$ after smoothing it at scale m.

Note that the edgels of $S_{i,h}$ are actually a subset of the edgels of $S_{i,l}$, but the super-segments themselves may not have this property, since edgels may be linked differently for different scales (at junctions, for example).

The hierarchical matching algorithm is as follows:

1. Match $S_{1,h}$ and $S_{2,h}$ to obtain a match $M(1,2,h) : S_{1,h} \Rightarrow S_{2,h}$ for the higher scale.
2. Match $S_{1,h}$ and $S_{1,l}$ to obtain a match $M(1,h,l) : S_{1,l} \Rightarrow S_{1,h}$ between the two scales for the first image.
3. Match $S_{2,h}$ and $S_{2,l}$ to obtain a match $M(2,h,l) = S_{2,h} \Rightarrow S_{2,l}$ between the two scales for the second image.
4. Combine the results to obtain *predicted matches* for the images smoothed by l, by applying the multiple-matches algorithm. Result: a match $S_{1,l} \Rightarrow S_{1,h} \Rightarrow S_{2,h} \Rightarrow S_{2,l}$. Remove the two middle matches to get the predicted matches: $P(1,2,l) : S_{1,l} \Rightarrow S_{2,l}$.
5. Use $P(1,2,l)$ to match $S_{1,l}$ and $S_{2,l}$ to obtain $M(1,2,l) : S_{1,l} \Rightarrow S_{2,l}$ for the lower scale.

Of course, steps 2 to 5 can be repeated for smaller scales.

In the first step we match a relatively sparse edgel map, since the image was smoothed with a very coarse scale filter. We therefore do not expect many competing matches, and the matching is relatively easy. The second and third steps are almost trivial, since the edgels shift by at most a pixel. We use our matching algorithm with a disparity of 2. The fourth step is a mere application of our multiple matches algorithm, described in [9] and also later in subsection 3.4. Although this method was originally designed for a motion sequence, it is easily applicable here. In the fifth step we use the predicted matches as guide for the matching algorithm in several steps. We describe this algorithm in the next subsection.

Example

Figures 1 and 2 show an example for the application of the hierarchical match. Subfigures 1(a) and (b) contain the two consecutive frames from a sequence of moving toy jeep and train. subfigures 1(c) and (d) contain the contours detected after smoothing with a mask of 16, subfigures 1(e) and (f) contain the contours detected after smoothing with a mask of 8 and subfigures 1(g) and (h) contain the contours detected after smoothing with a mask of 4.

Subfigure 2(a) contains the matches computed for a scale parameter of 16, subfigure 2(b) contains the predicted matches computed from mask 16 for a scale parameter of mask 8, subfigure 2(c) contains the matches computed for mask 8 using the predicted matches of previous step, subfigure 2(d) contains the predicted matches computed from a scale parameter of 8 to a scale parameter of 4 and subfigure 2(e) contains the matches computed for a scale parameter of 4 using the predicted matches of the previous step. These are the final matches for this pair of images. Finally subfigure 2(f) contains the multiple matches.

3.3 DESCRIPTION OF THE MATCHING ALGORITHM

In this section we describe our matching algorithm. Given two sets of super-segments detected in the two matched images, both smoothed with the same scale parameter, a set P of predicted matches computed by steps 1 to 4 of the hierarchical matching algorithm (see subsection 3.2) and the maximal disparity d, our intent is to find for each edgel contour section in one image, the matching section in the other image. Our matching criteria is *similarity* between the contour sections, which we define as the minimum area between them (after translating one of the sections to the beginning of the other) divided by the combined length (in edgels) of the two sections squared. We choose this similarity measure, since our goal is to both minimize the area between the two matching sections and yet give preferances to matching longer sections over short ones (that may have smaller area in between them). The longer the matching sections are, the better chance there is for the match to be correct.

In doing so, the following problems need to be addressed:

1. Initial contour match:

   Approximate each contour by a list of line segments and match them according to their position, orientation and strength. Two contours can match only if they have some segments matching. Further, the segment matches help localizing the contour matches later on.

   For each segment in the first image, search for possible matching segments in the second image around its location upto distance d. However, if a predicted match (from higher level) exists for this segment, use the prediction instead by searching only in a *small* (we used 5 pixels instead of d) window around the location of the predicted match, thus reducing search time as well as the chance for incorrect matches.

2. Partial contour match:

   Edgels may disappear or appear due to either physical causes such as occlusion or errors in the processing steps (edge detection, thresholding, linking). As a result, contours that actually belong to different objects may merge, or an object contour may break into several smaller disconnected contours or even partly disappear.

   Our solution is to divide the contours arbitrarily into small enough sections that will most probably not be larger than the true matching sections, yet long enough to convey meaningful information. Each such section is then compared to a possible matching contour and is slid along it for the best match. Results of the segment matching step are used to restrict the size of the matching contour section: if a section p_1 of a super-segement s_1 in $image_1$ is compared to a super-segment s_2, the matching section p_2 has to have the segments overlapping it match the segments overlapping p_1.

   Results of the segment matching can be used to determine maximal matching sections (by simply combining adjacent segment matches together). These maximal sections are then divided into smaller sections, each of which is then independently matched.

   Choosing the "right" size for these smaller sections is an issue here - too short a section will not yield a unique match. Too long a section may include more than one object. A constant section size will tend to penalize short matches and a constant number of sections per contour will tend to penalize long ones. Our solution is as follows: For every maximal section, any predicted match overlapping it is a section. The remainder of the maximal section (that may be quite fragmented after cutting out the predictions) is partitioned into sections of length approximating $\frac{2 \cdot l}{\log(l)}$, where l is the length of the shorter of the two maximal sections matched

   For each section p_1 in s_1 find the section of s_2 within which p_1 may match (use matches of p_1 segments to determine the range). Then slide p_1 along this section and look for the most similar match.

   If p_1 is section taken from a predicted match, this search is much simpler. Only matches that represent similar 2D translation to the predicted match can be selected.

   Once a possible match is detected, "extend" it by adding adjacent edgels as long as the similarity (previously defined) decreases. To reduce time complexity, this step can be done in a binary search type operation.

   Note that this step can create many competing matches: if a super-segment was divided into several sections, each of which matching correctly and then extended, we may get several nearly identical (and overlapping) matches.

3. Dealing with spurious matches

   So far we have described how to create matching contour sections. We define spurious matches as matches that are either incorrect or overlap other correct matches (and should therefore be eliminated).

   Incorrect matches very often correspond to random motion, and thus can be distinguished from correct matches. To detect such matches, we have designed a simple relaxation procedure. Every match is identified by the 2-

D translation it represents and its length. One match can *support* some other match if they have similar translation (that is, their translation differs by no more than some constant threshold). Very short matches are assumed arbitrary and cannot therefore be used to support other matches. A match is considered spurious (and therefore discarded) if the total length of matches supporting it is below some given constant threshold and none of these matches is long.

Overlapping matches are possibly correct matches that share sections or points. Either the sections from the first image overlap or from the other image or both. The sections may partially or completely overlap each other. In both cases we choose the better match (the more similar one) and the remainder of the other match. This step usually implies a significant drop in the amount of information we need to handle, since adjacent sections matched correctly and extended tend to create overlapping matches.

3.4 Combining pairwise matches

Matching only two images is not very useful for motion detection and estimation. Our method for combining pairwise section matches into multiple section matches is fairly simple.

If section P_1 (in frame 1) matches section P_2 (in frame 2) and section Q_2 (in frame 2) matches section Q_3 (in frame 3),then P_2 and Q_2 either have no points in common, one of them is a sub-section of the other or they partly overlap. Combining these matches is, of course, possible only in the last two cases. This problem is very similar to the overlapping of matches discussed previously. Our solution is to compute the overlapping sub-section of P_2 and Q_2, say R_2 and then find R_1, the sub-section of P_1 (in frame 1) that best matches R_2. R_3 is computed in a similar way. The result is a match (R_1, R_2, R_3). This process can be iteratively applied to obtain multiple matches $(M_1, M_2, \ldots, M_k)$ for k frames. To eliminate bad multiple matches (if a pair of matches was erronous, the whole multiple match is incorrect), the variance in the 2-D translations between successive frames is thresholded.

We applied this simple algorithm to the sequences in the results section and it performed well.

Problems with this method are mainly that it can only handle sections that match throughout the sequence. Disappearance of points (due to occlusion or disappearance of objects from the images) cannot be handled by this simple algorithm. In addition, this method will discard a long multiple match if one of the matching pairs is wrong (and so the variance between matches is large). It would be better to detect this error instead. We are working now on extending the method to handle these problems.

4 RESULTS

We have applied our algorithm to a number of real images, indoor as well as outdoor scenes. As long as the shapes of objects in the scene (as projected in the image) does not change significantly, the results are very good. We show them by displaying only those points for which a match was found, and drawing an arrow to the closest point in the other section (after translating to start at same location). The arrow is drawn for every fifth point in a matching section (for each matching section), for clarity.

We give three examples, the matching for each was done using scale parameters of 16, 8 and 4. We show only the multiple matches for the smallest scale.

(a) Figures 3(a),(b) and (c) show two 240 × 240 pixels images taken from a sequence of a road scene. Both the observer and the other car are moving. We have selected every 12th frame for the matching. The disparity is about 30 pixels.

(b) Figures 3(d),(e) and (f) show two 256 × 256 images of a corridor. The camera faces the direction of motion, so we expect objects to expand. The matched images are spaced 10 frames apart. The disparity is about 20 pixels.

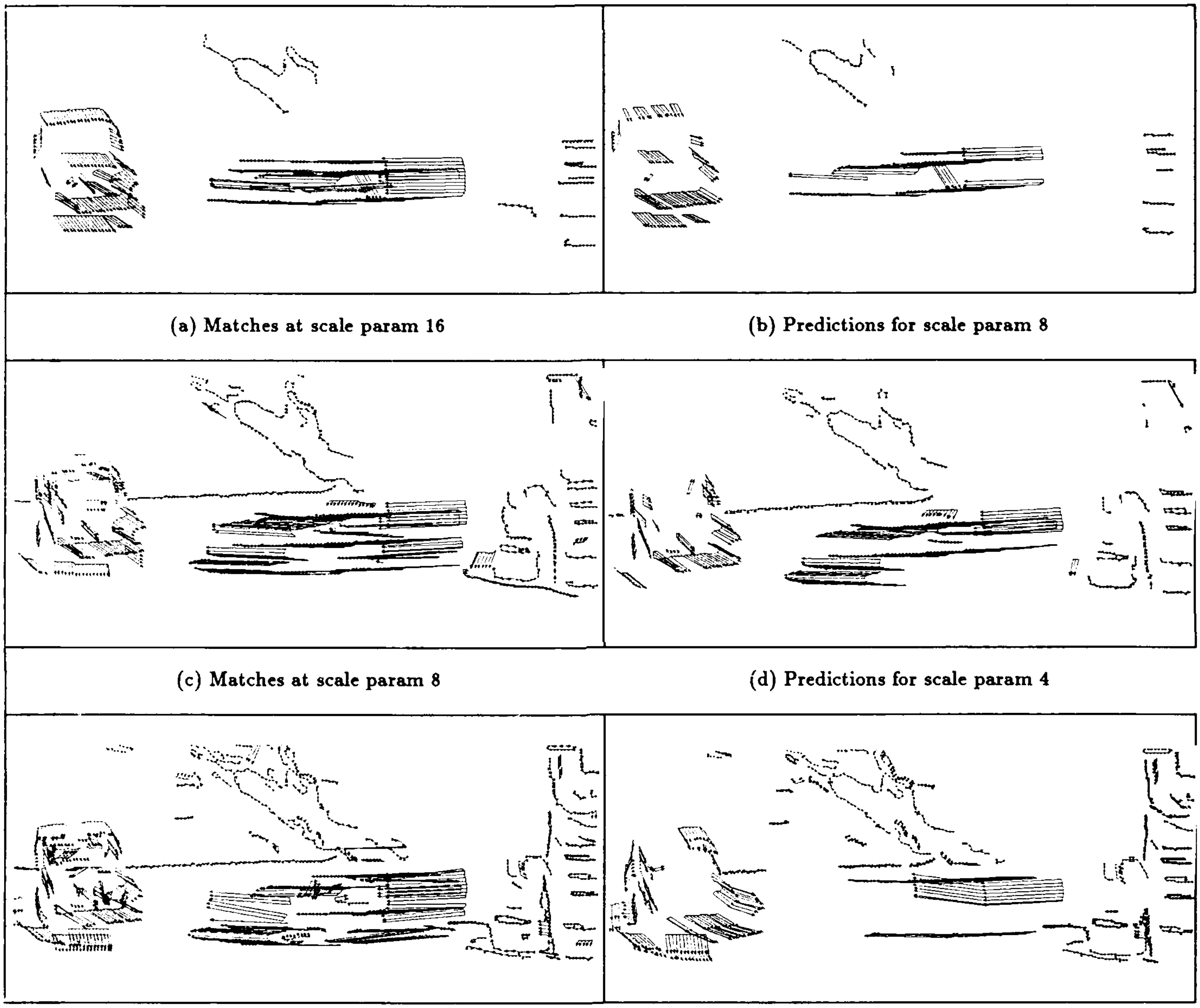
(c) Figures 1(a) and (b) contain two 250 × 512 images of a toy jeep and train. The camera is stationary, but both the jeep and the train are moving. This is a difficult scene as the motion is both very large (disparity of at least 70) and is a general 3D motion. These figures were described in detail in subsection 3.2.

The images in Figure 3 were obtained from SRI International, courtesy of Dr. Bolles.

The program to implement our algorithm was written in Common LISP on a *Symbolics* Lisp Machine.

5 CRITICAL EVALUATION

We have shown an algorithm to compute correspondences between 2 frames with very few constraints. We suggested the use of edgel contours and sections of contours. Correspondence was based on shape similarity



(a) Matches at scale param 16

(b) Predictions for scale param 8

(c) Matches at scale param 8

(d) Predictions for scale param 4

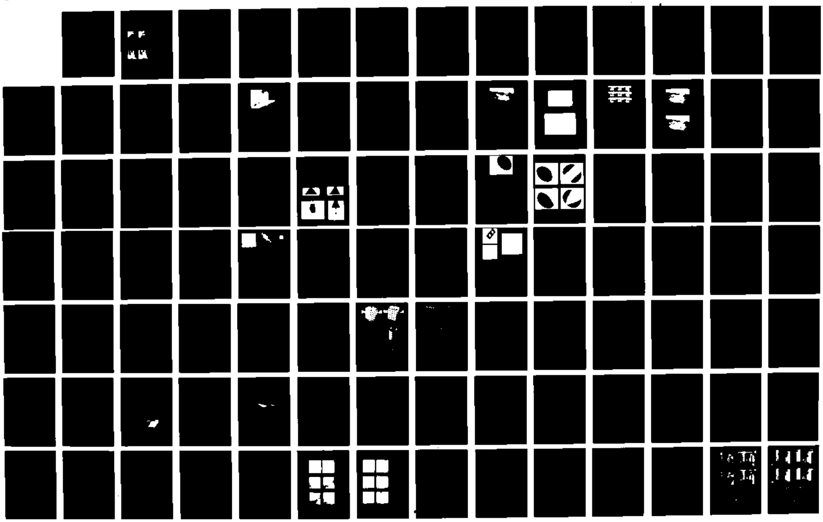
(e) Matches at scale param 4 (final)

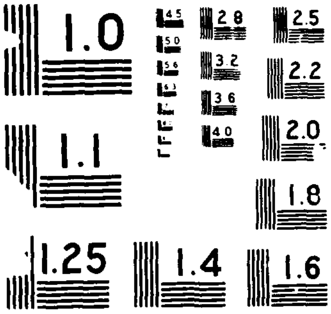
(f) Multiple matches for scale param 4

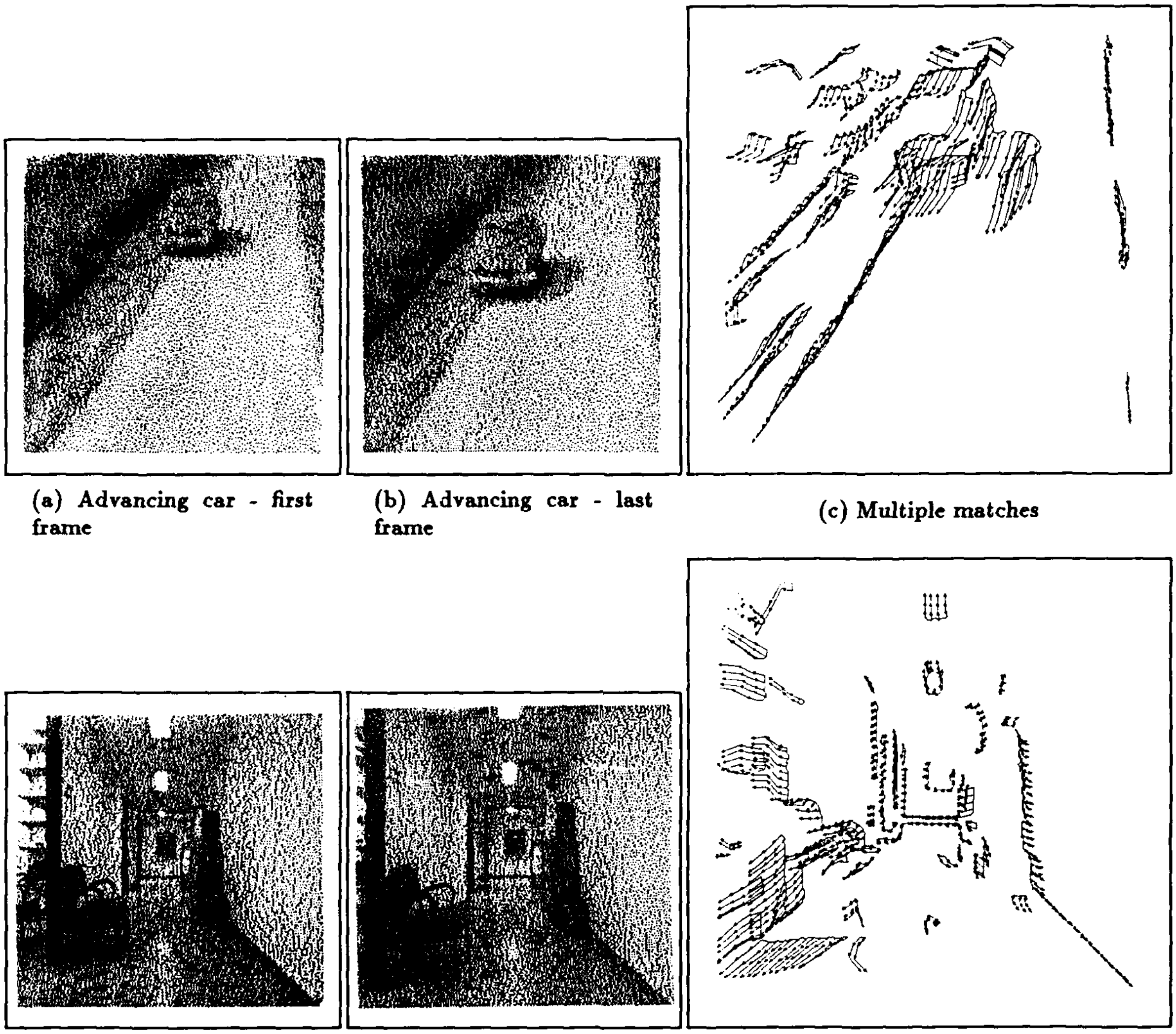
Figure 2: Jeep and train sequence - execution of the algorithm

AD-A212 981 PROCEEDINGS OF IMAGE UNDERSTANDING WORKSHOP HELD IN 11/13
PALO ALTO CALIFORNIA O. (U) SCIENCE APPLICATIONS
INTERNATIONAL CORP MCLEAN VA L S BAUMAN MAY 89

UNCLASSIFIED N00014-86-C-0700 F/G 17/11 NL




1.0
4.5
5.0
5.6
6.3
2.8
3.2
3.6
4.0
2.5
2.2
2.0
1.1
1.8
1.25
1.4
1.6



(a) Advancing car - first frame

(b) Advancing car - last frame

(c) Multiple matches

(d) Hallway - first frame

(e) Hallway - last frame

(f) Multiple matches

Figure 3: Multiple matches for the advancing car and Hallway sequences

between matching sections and on translation similarity between matches, and demonstrated some results on a number of real images. Using a multi-scale approach enables us to use better primitives.

The advantages of the matching method were discussed in the previous sections: the use of continuity and sections of arbitrary shape and size in matching, the use of length of a match, evaluation of matches based both on shape and on common translation. The advantages of adaptive filters are in their better localization of edgels, edgels that represent real events, and their hierarchical nature. The advantages of using multi-scale hierarchical matches are: reduced complexity and the ability to match well even with smaller masks.

Applying this method to real images produces very good results. We have been able to match less closely located frames yet obtain better results than with previous approaches.

Some comments are in order:

- The algorithm has a very heuristic flavor.
- Our multiple matches algorithm 3.4 only keeps those matches that extend throughout the sequence, and thus cannot handle appearance and disappearance of points.
- The computation is made in 2-D only, but we can find the actually corresponding points using areas of high curvature or even the simple method we used for displaying the results (a left point matches the closest point in the translated matching right section). These point-to-point matches can be used for motion estimation in 3-D. We are currently working on using the Motion Estimation algorithm developed in [19]. This algorithm uses matching points in three or more frames to estimate 3-D motion and location of points in frames as well as give some error measure to the match. Since using a Motion Estimation algorithm requires matches in multiple frames, an algorithm to combine the results of matching pairs of images will be useful.

Rcferences

[1] J. K. Aggarwal and R. C. Duda. Computer analysis of moving polygonal images. *IEEE Transactions on Computers*, C-24(10), 1975.

[2] M. Asada, M. Yachida, and S. Tsuji. Three dimensional motion interpretation for the sequence of line drawings. In *Proceedings of the International Conference on Pattern Recognition*, pages 1266–1273, Dec. 1980.

[3] N. Ayache and B. Faverjon. A fast stereovision matcher based on prediction and recursive verification of hypothesis. In *Proceedings of the 3rd IEEE Workshop on Computer Vision: Representation and Control*, pages 27–37, Bellaire, Michigan, Oct. 1985.

[4] J. Canny. A computational approach to edge detection. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, PAMI-8(6):679–698, Nov. 1986.

[5] J. S. Chen and G. Medioni. Detection, localization and estimation of edges. In *Proceedings of Workshop on Computer Vision*, Miami Beach, Florida, Nov. 1987. IEEE.

[6] L. S. Davis, Z. Wu, and H. Sun. Contour-based motion estimation. *Computer Vision, Graphics and Image Processing*, 23:313–326, 1983.

[7] L. Dreschler and H.-H. Nagel. Volumetric model and 3-d trajectory of a moving car derived from monocular tv-frame sequence of a street scene. In *International Joint Conference on Artificial Intelligence*, Vancouver, Canada, Aug. 1981.

[8] S. L. Gazit and G. Medioni. Accurate detection and linking of zero crossings. Unpublished, 1987.

[9] S.L. Gazit and G. Medioni. Contour correspondences in dynamic imagery. In *Proceedings of the Image Understanding Workshop*, volume 1, pages 423–432, April 1988.

[10] D. B. Gennery. Ananlysis stereo system for analysis autonomous vehicle. In *Proceedings of the Fifth International Joint Conference on Artificial Intelligence*, pages 576–582, Aug. 1977.

[11] F. Glazer, G. Reynolds, and P. Anandan. Scene matching by hierarchical correlation. In *Proceedings of the Conference on Computer Vision and Pattern Recognition*, pages 432–441, June 1983.

[12] C. J. Jacobus, R. T. Chien, and J. M. Selander. Motion detection and analysis by matching graphs of intermediate level primitives. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, PAMI-2(6):495–510, Nov. 1980.

[13] R. Jain, D. Militzer, and H.-H Nagel. Separating non-stationary from stationary scene components in a sequence of real world tv-images. In *Proceedings of the 5th International Joint Conference on Artificial Intelligence*, pages 612–618, Cambridge, Mass, Aug. 1977.

[14] H. S. Lim and T. O. Binford. Structural correspondence in stereo vision. In *Proceedings of Image Understanding Workshop*, pages 794–808, 1987.

[15] D. Marr and T. Poggio. A theory of human stereo vision. Memo 451, Massachusettes Institute of Technology, Nov. 1977.

[16] G. Medioni and R. Nevatia. Segment-based stereo matching. *Computer Vision, Graphics, and Image Processing*, 31:2–18, 1985.

[17] K. Price and R. Reddy. Matching segments of images. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, PAMI-1(1):110–116, Jan. 1979.

[18] P. Saint-Marc and G. Medioni. Adaptive smoothing for feature extraction. In *Proceedings of Image Understanding Workshop*, pages 1100–1113, Boston, Mass, Apr. 1988.

[19] H. Shariat. *The Motion Problem - How to use more than two frames.* PhD thesis, IRIS, University of Southern California, Los Angeles, California, Oct. 1986.

[20] M. H. Singer. Significant feature detection and matching in image pairs. In *Proceedings of the Tenth International Joint Conference on Artificial Intelligence*, pages 829–831, Aug. 1987.

[21] W. E. Snyder. Computer analysis of time varying images. *Computer*, 14(8):7–9, Aug. 1981.

[22] S. Ullman. *The Interpretation of Visual Motion.* MIT Press, Cambridge, Mass, 1979.

[23] A. P. Witkin. Scale-space filtering. In *Proceedings of the International Joint Conference on Artificial Intelligence*, pages 1019–1022, Karlsruhe, Germany, 1983.

Representations in High-Level Vision: Reassessing the Inverse Optics Paradigm

Shimon Edelman and Tomaso Poggio
Artificial Intelligence Laboratory, MIT, Cambridge, MA 02139

1 Inverse optics

What performance is it reasonable to require from an artificial vision system before it can be properly regarded as seeing? Paraphrasing Marr, one would expect such a system to know what is there by looking ([20], p.3). This appears to imply that (1) the system must have a way to represent what is there in some fashion and (2) this representation must be effectively computable (in fact, it should better be *efficiently* computable, given the system's resources).

Marr stressed that the choice of representation is important because it can greatly influence the complexity of any subsequent processing of the information that is being represented. Thus, if intensity edges are used by some processing stage in the system, they should be made explicit in the representation computed by an earlier stage. In higher-level visual tasks such as scene understanding the choice of an output representation becomes less obvious if not downright arbitrary. For example, it appears to depend on the end to which the scene representation is the means: an "analog", map-like representation of a room would be useful for navigation but superfluous for merely finding out whether a specified object is present in the room (cf. [35]).

In object recognition (if it is not part of a navigation strategy), the computed representation must serve two related purposes. The first is identification: the representation must contain enough information to allow the system to choose from a set of known objects one that is most similar to the attended object, or to declare with a certain degree of confidence that the latter has not been previously encountered. The second purpose is that of association: the representation must allow access to additional information associated with the recognized object. This could be visual (e.g. what does the opposite side of the object look like), or general (e.g. its weight).

The idea of association is to rely on past experience to gain access to more information than is actually present in the visual input. Realizing that efficient association is, after all, the main objective of vision helps to define the requirements of an intelligent system from its vision module. The main conclusion we draw from this realization is that detailed geometrical representation of objects and scenes may not be necessary for recognition.

We illustrate this point by the simplified example in Figure 1. An object fragment such as the one that appears there may remind our system of a cat. In fact, if it does not remind it of much more than that, then it probably *is* a usable representation of a cat. Several such representations, each useful under different circumstances, may be kept by the system.

In real-world scenes, as opposed to Figure 1, abundant visual information is available. Most of this information is irrelevant for creating a "first-order" interpretation of the scene. Even if the system can select for further processing only those parts of the image that seem a priori most important, too much

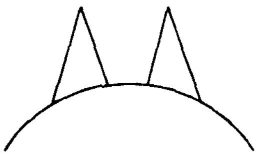


Figure 1: Two pointed things, one beside the other, on top of something rounded... Could be a cat's ears?

information may be still there that is both difficult to extract and unnecessary for recognition. For example, the knowledge (supplied by the stereo module) which of the two Λ-like features detected in the image is farther from the cameras does not seem to increase or diminish the probability that the two Λs are in fact a cat's ears. Qualitative information, such as an assertion that the two Λs are several centimeters rather than several meters apart in depth, could be easier to compute and no less useful for recognition.

Difficulties with computing exact representations, and arguments similar to the one outlined above, have led in the past, on one hand, to feature-based approaches to recognition in which objects were represented by lists of values of several (often, many) perceptual variables [7], and, on the other hand, to top-down strategies that invoked general knowledge about the scene to assist object recognition [21]. However, Rosenfeld [28] pointed out that a sophisticated visual system should be able to recall by itself from memory the frame within which an image is to be interpreted (or to construct such a frame if a novel coincidence of an object persistently appearing in a certain context is noticed). Furthermore, recognition based on the invariant feature approach appeared to be infeasible, even when top-down constraints were invoked. Partly because of that, the paradigm of computer vision that emerged during the last decade calls for achieving representations through inverse optics. Its central tenet is that objects are best recognized only after their visible surfaces have been described geometrically, e.g. as sets of points in some (preferably, object-centered) coordinate system [20]. The inverse optics paradigm – as put forward by [?] – claims that physical properties of the surfaces must be computed from the images, but not necessarily everywhere or very precisely. In the sequel, we argue in fact that vision does not necessarily require *exact* representations. Accordingly, we see the goal of inverse optics in the derivation of the physical properties of 3D surfaces at a qualitative, not very precise level, as suggested by human vision.

A recent review of object recognition methodologies by Ullman [37] distinguishes three main classes of theories: (1) invariant properties methods, (2) parts decomposition methods, and (3) alignment methods.

> Theories in the first class assume that that certain simple properties remain invariant under the transformations that an object is allowed to make [such as the transformation arising from viewing a 3D object from different points in space]. This approach leads to the notion of invariances, feature spaces, clustering, and separation techniques. The second class relies on the decomposition of objects into parts. This leads into the notions of symbolic structural descriptions, feature hierarchies and syntactic pattern recognition. By and large, the first of these general approaches was the dominant one in the earlier days of pattern recognition and the second approach has become more popular in recent years.

Ullman argued that both these approaches are insufficient for the general problem of shape-based visual recognition. Instead, he proposed a method, called alignment, that involves more or less exact pictorial representations of object models. In Ullman's approach, a small number of corresponding key features in the image and in a model are used to compute the transformation that would bring the two into alignment. After normalization by alignment, the two shapes are compared to decide whether the image is actually an instance of the model.

2 A possible alternative

The main shortcoming of the invariant features and the structural decomposition approaches appears to be brittleness. It is difficult to find a set of features each of which is both simple enough to be computed reliably by a low-level process and sufficiently invariant over all possible views of an object. Thus, pure feature-based methods tend to fail when presented with scenes in which many similar objects may appear in widely varying poses. On the other hand, structural decomposition has difficulties with achieving stable descriptions of many real-world objects (such as a shoe or a rabbit, cf. [37]). In addition, this approach by definition must recover the three-dimensional structure of the image, to compare it with structurally specified object models, that is, it involves inverse optics.

Alignment of pictorial descriptions and similar schemes ([19], [14], [34] [17]) use 3D or $2\frac{1}{2}$D models but do not require inverse optics. Instead, alignment calls for computing the (2D) appearance of each model from the viewpoint constrained by key feature location in the image and for comparing that with the image. These methods appear to work under a variety of conditions. However, when applied to articulated or flexible objects, alignment had to be combined with some kind of structural decomposition. For example, it has been proposed to compute separate aligning transformations for different regions of articulated objects [37], and an alignment-based system for handwriting recognition had to decompose letters into smaller parts (strokes) to compensate for the variability of letter structure [9].

We interpret these compromises as arguments in favor of an approach to the representation problem that is more qualitative in that it is restricted neither to vectors of feature values, nor to syntactic constructs in a formal language, nor to pictorial descriptions. This approach combines elements of the three methods distinguished by Ullman and may be termed linguistic because it employs what looks like natural language descriptions of objects. It may also be called open-ended, because it does not prescribe a standard formula for representing different things, but rather leaves that to be determined by the interaction of the system with its environment.

2.1 Open-ended representations

Is there a good way to describe an object (say, a cat) to a person who never saw one, but who possesses much the same vocabulary as you do? There might be, if it is possible to summarize the differences between a cat and other, familiar, objects (say, dogs) that are similar to it in a few words. An example could be, "a cat looks like a dachshund, except that it has sort of pointed ears, its legs are straighter and longer," etc. This approach presupposes prior knowledge of somewhat complicated concepts such as an ear or a leg, but does not normally require exact decomposition of the object into parts corresponding to these concepts (there is no need to define the precise boundary between the head and the ear to see that (i) there is something like an ear on top of the head and (ii) it is pointed). It uses feature values (*pointed* ears) to distinguish between similar objects, but defines the features locally (pointed *ears*) rather than attempting to find out the range of the ratio of area to perimeter that is characteristic of an image of a cat. In some cases, this approach must resort to pictorial descriptions ("pointed like this: Λ" — see Ullman [37], p.18, for a discussion of the need of detailed shape description in recognition). These descriptions, however, would also be local and might often be aimed at capturing ill-localized properties that are difficult to describe verbally, such as texture.

2.2 Synthesis

We propose to integrate the three approaches to representation (feature spaces, and structural and pictorial descriptions) by adopting the following as a basic set of requirements from a representation scheme:

- **Grounding in robust low-level representations.** The representation of objects and scenes should be based on maps of visual space that integrate different cues such as intensity, texture, stereo, and motion ([25], cf. the notion of base representations [36]).

- **Sparseness.** Objects should characterized by a small number of tags or features that come from a very large repertoire. A convenient analogy is description by short sentences in a natural language (cf. [27]).

- **Similar treatment objects and of static and dynamic scenes.** By further analogy, situations may be described using the same simple language. Dynamic situations (happenings and actions) may be distinguished by tags that have verb-like connotations.

- **Localized features.** The features should be two-dimensional (that is, detectable in 2D views, without depth recovery) and spatially localized (cf. [4]). The localization could be coarse and may employ relations computed by universal visual routines [36].

- **Shallow hierarchy.** Although nontrivial objects, themselves composed of several features, could in turn serve as features in a more complicated representation, the hierarchy should be shallow, e.g. *cat* to *ears* to low-level place tokens marking intensity edges, texture etc.

- **Heterogeneity and flexibility.** The scheme should not prescribe the precise format of or the amount of information present in the representations of various objects. Rather, it should be demand-driven and augment existing representations when the need arises for finer distinctions among some of the objects.

- **Robustness through representing similarity.** A system confronted with a novel object should come up with a plausible first guess (the "nearest neighbor" of the unknown object). This could be facilitated by maintaining an explicit representation of similarity relations among the feature sets that define objects.

- **Incremental learnability.** The scheme should allow the system to bootstrap itself by learning representations for novel objects, given only low-level routines (the apparatus for computing the base representations) and real-world input. Furthermore, learning should not require an oracle capable of specifying the desired representation for a given input. Self-organizing representations satisfying this constraint have been demonstrated for some low-level visual features ([18], [29]).

3 Discussion

The trend of reassessing the commitment to inverse optics can be discerned in the proceedings of a recent conference on computer vision. Some examples of this are mentioned below.

- The stress on obtaining qualitative rather than quantitative solutions to problems in vision (qualitative depth from stereo [39], qualitative description of motion for obstacle avoidance [22], see also [38]);

- Development of object representations that are 2D (or $2\frac{1}{2}$D) and viewer-centered, rather than 3D and object-centered (aspect graphs [12], [15], [33], [32]; a novel curvature-based representation [2]);

- Increasing interest in active vision (using egomotion information [13], attentive gaze control [6]);

- Efforts to integrate multiple sensors/cues (a general strategy for integration [15], integration of focus, vergence and stereo [1]).

In the domain of visual motion study, Thompson and Kearney [35] have argued recently in favor of "inexact" or qualitative vision. Their definition of a qualitative representation is as follows:

> If an attribute can take only a small number of values we say that a representation is qualitative. Qualitative representations define a set of equivalence classes on the quantitative scale. We will assume qualitative values correspond to disjoint, continuous intervals on the quantitative scale.

In other words, qualitative vision is equated with low-resolution vision, leaving the problem of (approximate) inverse optics in its place. Starting from an observation made by Thompson and Kearney, namely, that a representation should contain only as much information as is necessary for subsequent processing, we have attempted to revise the strict (and mistaken) form of the inverse optics paradigm. However, rather than abolishing the notion of inverse optics (along with the notion of representing the visual world in the usual sense; see [5]), we propose to confine it to the low-level processing responsible for the formation of integrated base representations.

3.1 Research directions

Exploration of the approach to visual knowledge representation that we have sketched above could proceed in four main directions. Development of robust algorithms for low-level vision, whose purpose is to compute the base representation, or the collection of feature maps, is perhaps the most important open problem. Some recent advances in this direction include the work on perceptual grouping ([19], [16] bottom-up detection of salient structures [30], integration of diverse visual cues, and detection and labeling of discontinuities in the resulting map ([24], [25], [11]).

The second class of problems, which may be described as middle vision, is centered around further investigation of visual routines [36] that could be used to link low-level features. In particular, a suitable control structure has to be developed to guide the application of the routines in accordance with the host system's goals and subject to environmental pressure and constraints. This issue is related to that of active vision [22]: strategy employed by a system engaged in active exploration of its environment is likely to differ from that of a passive observer.

The third domain of exploration, traditionally referred to as high-level vision, includes improving recently developed recognition methods ([19], [14], [34]) and testing their limitations. In particular, approaches based on exact geometrical representation (such as the various alignment methods) may be subsumed by a general, qualitative scheme, which views recognition as a three-stage process [8]. In the first stage of this process, the system selects from its input a region of interest that would be subjected to further processing. Such a region may be defined by motion boundaries, structural saliency [30] and other information directly computable from the base representation. Next, an indexing stage could be introduced into the recognition process to narrow down the set of object models that would participate in a subsequent matching, or verification by alignment. Finally, instead of being precisely aligned with the image, an object model could be roughly rotated so that the image and the model are seen from the same general aspect. This coarse alignment would be followed by a qualitative comparison of the visible features of the kind discussed above. Alternatively, new verification methods could be devised that rely on processes other than the transformation of object models. All of the above suggestions amount to a reassessment of the issue of object representation, without which, we believe, little progress is to be expected in extending state of the art recognition techniques to real-world images.

Finally, work on knowledge acquisition and representation must be brought to bear on the issue of visual representation, if the latter is myto be considered in the context of general intelligent behavior. With the outline of section 2.2 in view, some topics that appear particularly relevant are semantic nets [10], similarity-based reasoning [31], efficient associative memory ([23], [3]) and decision trees [26].

3.2 Summary

We have outlined an approach to the representation of objects and situations for visual recognition that combines detailed, integrated feature maps computed in a bottom-up fashion by low-level processes with qualitative, heterogeneous, open-ended high-level representations. Whether this approach will be more successful than the current paradigm that considers vision by and large as inverse optics is an empirical issue that remains to be settled.

References

[1] A. Abbott and N. Ahuja. Surface reconstruction by dynamic integration of focus, camera vergence and stereo. In *Proceedings of the 2nd International Conference on Computer Vision*, pages 532–545, Tarpon Springs, FL, 1988. IEEE, Washington, DC.

[2] R. Basri and S. Ullman. The alignment of objects with smooth surfaces. In *Proceedings of the 2nd International Conference on Computer Vision*, pages 482–488, Tarpon Springs, FL, 1988. IEEE, Washington, DC.

[3] E. Baum, J. Moody, and F. Wilczek. Internal representations for associative memory. *Biological Cybernetics*, 59:217–228, 1988.

[4] R. Bolles and R. Cain. Recognizing and locating partially visible objects: the local feature focus method. *International Journal of Robotics Research*, 1:57–82, 1982.

[5] R. Brooks. Intelligence without representation. In *Proc. of Foundations of AI Workshop*, MIT, June 1987.

[6] J. Clark and N. Ferrier. Modal control of an attentive visual system. In *Proceedings of the 2nd International Conference on Computer Vision*, pages 514–523, Tarpon Springs, FL, 1988. IEEE, Washington, DC.

[7] R. Duda and P. Hart. *Pattern classification and scene analysis*. Wiley, New York, 1973.

[8] S Edelman and T. Poggio. Integrating visual cues for object segmentation and recognition, 1989. in press.

[9] S. Edelman and S. Ullman. Reading cursive script by computer, May 1989. to appear in 42nd SPSE Conference Proceedings.

[10] S. Fahlman. *NETL: a system for representing and using real-world knowledge*. MIT Press, Cambridge, MA, 1979.

[11] E. Gamble, D. Weinshall, D. Geiger, and T. Poggio. Labeling edges and the integration of low-level visual modules, 1989. in preparation.

[12] Z. Gigus, J. Canny, and R. Seidel. Efficiently computing and representing aspect graphs of polyhedral objects. In *Proceedings of the 2nd International Conference on Computer Vision*, pages 30–39, Tarpon Springs, FL, 1988. IEEE, Washington, DC.

[13] D. Heeger and G. Hager. Egomotion and the stabilized world. In *Proceedings of the 2nd International Conference on Computer Vision*, pages 435–440, Tarpon Springs, FL, 1988. IEEE, Washington, DC.

[14] D. Huttenlocher and S. Ullman. Object recognition using alignment. In *Proceedings of the 1st International Conference on Computer Vision*, pages 102–111, London, England, June 1987. IEEE, Washington, DC.

[15] K. Ikeuchi and T. Kanade. Applying sensor models to automatic generation of object recognition programs. In *Proceedings of the 2nd International Conference on Computer Vision*, pages 228–237, Tarpon Springs, FL, 1988. IEEE, Washington, DC.

[16] D. Jacobs. The use of grouping in visual object recognition. TR 1023, MIT, January 1988.

[17] Y. Lamdan and H. Wolfson. Geometric hashing: a general and efficient recognition scheme. In *Proceedings of the 2nd International Conference on Computer Vision*, pages 238–251, Tarpon Springs, FL, 1988. IEEE, Washington, DC.

[18] R. Linsker. Self-organization in a perceptual network. *IEEE Computer*, 21:105–117, March 1988.

[19] D. G. Lowe. *Perceptual Organization and Visual Recognition*. Kluwer Academic Publishers, Boston, 1986.

[20] D. Marr. *Vision: A Computational Investigation into the Human Representation and Processing of Visual Information*. W.H. Freeman and Company, San Francisco, 1982.

[21] M. Minsky. A framework for representing knowledge. In P. Winston, editor, *The psychology of computer vision*. McGraw-Hill, New York, 1975.

[22] R. Nelson and J. Aloimonos. Using flow field divergence for obstacle avoidance: towards qualitative vision. In *Proceedings of the 2nd International Conference on Computer Vision*, pages 188–196, Tarpon Springs, FL, 1988. IEEE, Washington, DC.

[23] S. M. Omohundro. Efficient algorithms with neural network behavior. UIUCDCS R-87-1331, Univ. of Illinois at Urbana-Champaign, April 1987.

[24] T. Poggio. *Integrating vision modules with coupled* MRF's. Working Paper No. 285, Artificial Intelligence Laboratory, Massachusetts Institute of Technology, 1985.

[25] T. Poggio, E. B. Gamble, and J. J. Little. Parallel integration of vision modules. *Science*, 242:436–440, 1988.

[26] J. R. Quinlan and R. Rivest. Inferring decision trees using the minimum description length principle, 1987. manuscript.

[27] W. Richards. How to play twenty questions with nature and win. A.I. Memo No. 660, Artificial Intelligence Laboratory, Massachusetts Institute of Technology, Dec. 1982.

[28] A. Rosenfeld. Recognizing unexpected objects: a proposed approach. *Int. J. of Pattern Recognition and Artificial Intelligence*, 1:71–84, 1987.

[29] T. Sanger, 1989. in preparation.

[30] A. Sha'ashua and S. Ullman. Structural saliency: the detection of globally salient structures using a locally connected network. In *Proceedings of the 2nd International Conference on Computer Vision*, pages 321-327, Tarpon Springs, FL, 1988. IEEE, Washington, DC.

[31] C. Stanfill and D. Waltz. Toward memory-based reasoning. *Communications of the ACM*, 29:1213–1228, 1986.

[32] L. Stark, D. Eggert, and K. Bowyer. Aspect graphs and nonlinear optimization in 3-d object recognition. In *Proceedings of the 2nd International Conference on Computer Vision*, pages 501–507, Tarpon Springs, FL, 1988. IEEE, Washington, DC.

[33] J. Stewman and K. Bowyer. Creating the perspective projection aspect graph of polyhedral objects. In *Proceedings of the 2nd International Conference on Computer Vision*, pages 494–500, Tarpon Springs, FL, 1988. IEEE, Washington, DC.

[34] D. Thompson and J. Mundy. Three-dimensional model matching from an unconstrained viewpoint. In *Proceedings of IEEE Conference on Robotics and Automation*, pages 208–220, Raleigh, NC, 1987.

[35] W. Thompson and J. Kearney. Inexact vision. In *Workshop on motion, representation and analysis*, pages 15–22, 1986.

[36] S. Ullman. Visual routines. *Cognition*, 18:97–159, 1984.

[37] S. Ullman. An approach to object recognition: Aligning pictorial descriptions. A.I. Memo No. 931, Artificial Intelligence Laboratory, Massachusetts Institute of Technology, Dec. 1986.

[38] A. Verri and T. Poggio. Against quantitative optical flow. In *Proceedings of the International Conference on Computer Vision*, pages 171–180, London, England, June 1987. IEEE, Washington, DC.

[39] D. Weinshall. Application of qualitative depth and shape from stereo. In *Proceedings of the 2nd International Conference on Computer Vision*, pages 144–148, Tarpon Springs, FL, 1988. IEEE, Washington, DC.

VISUAL RE-ACQUISITION OF GEOGRAPHIC LOCATIONS

Tod S. Levitt
Advanced Decision Systems
1500 Plymouth Street
Mountain View, CA 94043

Daryl T. Lawton
Advanced Decision Systems
1500 Plymouth Street
Mountain View, CA 94043

1. INTRODUCTION

Potential tasks for robots navigating in unknown or partially known environments include exploration of hostile environments such as Antarctica, the sea floor, other planets and/or asteroids; providing assistance in natural disasters such as floods, tornadoes or earthquakes; military surveillance, and trouble-shooting in industrial accidents such as mine cave-ins or nuclear melt-downs. All of these tasks have in common that the environment can change from what was previously known, but are likely to retain many landscape features.

The basic scenario we consider for the robotic use of visual memory is that the robot has stored memories from a previous trip through a geographic area, and is beginning another exploration of a different geographic area that shares visibility of landmarks viewed during the first trip. As the robot proceeds on his journey, this relationship between currently perceived visual events and memories allows the robot to build up a visual map of the environment. We can think of exploration as a process of tracking back and forth over an area, on each successive pass keeping visual contact with landmarks acquired on the last pass. Of course, visual memory can also be used to navigate back to a previously visited location.

The theory of qualitative navigation developed by Levitt and Lawton [Levitt et al. 87(a,b,c), 88] extended and quantified Kuipers' [Kuipers 77, 78, 82][Kuipers and Levitt - 88] original definition of a visually defined, distinct physical place in the 3D geographic world. The notion of a place as a viewframe is a circular panorama of visual landmarks stored in visual memory. A robot roams about the world, building up its visual memory of sequences of viewframes, i.e. viewpaths, making range and angle estimates as it goes. The theory already developed gives answers for planning and execution given reacquisition of some landmarks. However it does not explain how to recognize or match landmarks as they are actually described by a vision system, i.e. at coarse levels of the is-a hierarchy.

This paper addresses the problem of re-acquiring a viewframe, given that the robot can see some of the landmarks, that the visual descriptions are at coarse levels of the is-a hierarchy, and that the robot is not necessarily in the same location of the world as it was when it first acquired the viewframes.

2. VIEWFRAME CREATION

A viewframe is a data structure that encodes a circular list of robot-perceived landmarks, the angular displacement between (circularly) adjacent landmarks, and a closed interval of range estimates for each landmark. As a representation of cognitive visual memory, the viewframe is sensor (i.e. ego) centered. However its use for localization of the robot is performed by forming a local coordinate system based on the landmarks, rather than a sensor-centered coordinate system. For details, see [Levitt et al. - 1987a].

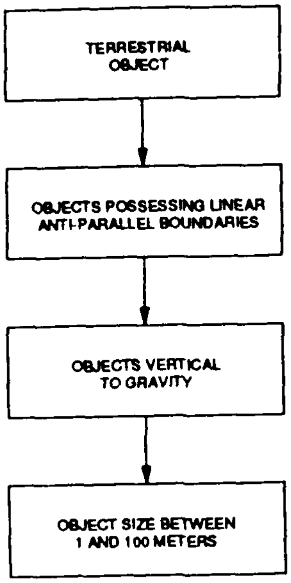

TERRESTRIAL OBJECT
OBJECTS POSSESSING LINEAR ANTI-PARALLEL BOUNDARIES
OBJECTS VERTICAL TO GRAVITY
OBJECT SIZE BETWEEN 1 AND 100 METERS

Figure 1: Landmark IS-A Hierarchy

Localization depends on the range estimates to landmarks, which can be extremely poor. However the robustness of the localization lies in the orthogorality of intervals of triangulations induced by landmark pairs. Range estimates are bounded by assuming that landmarks of interest will have height bounded between known values. Back-Projection from the pixel height then yields a worst-case range interval. Reasonableness of this height assumption depends on the criteria that are used to extract landmarks from imagery and how accurately this criteria models reality.

We define landmarks according to the partial is-a hierarchy of Figure 1. Inferring that an object is terrestrial requires locating the ground plane local to the object. The robot can carry a gravity sensor, or verticality may be inferred from an observed horizon line. Verticality restricts the search for anti-parallel lines, as does the requirement that the boundaries be adjacent to the ground plane. With the worst case range bounds, site is not a significant search restriction, however it is a stored attribute that can serve to distinguish viewframe landmarks. In addition to meeting perceptual criteria defined by the hierarchy, landmarks are described by a coarse color attribute. See [Lawton, Levitt, and Gelband - 88] for an example of automated viewframe extraction based on this perceptual description.

Key assumptions regarding a robot for qualitative navigation are

1. It has a wide field of view. A complete panoramic view would be optimal. This could be approximated by several cameras with different directions of view.
2. It has access to the position and orientation of the immediate ground plane. The robot has access to the position of it's "feet" or wheels to estimate the orientation of the immediate ground plane on which

it is standing and to its orientation relative to the direction of gravity.

3. Camera motion is restricted to being translational as the robot moves from place to place. This assumption isn't necessary but it does simplifies the motion processing that is required, especially for the extraction of occlusion boundaries. Realistically, the motion is required to be only roughly translational. Rotational parameters or effects or camera jitter could be detected by sensors (notably fiber-optic rotation sensors) and their effects either compensated for or the corresponding images removed from processing.

The robot initially extracts interesting perceptual groups from the panoramic images to form an initial viewframe. These groups correspond to 2D patterns which are recognized in images for such things as straight lines aligned with gravity, brightly colored patches which are different than surrounding patches, anti-parallel line sets, non-inflected curves, T-junctions, and points of high curvature.

As the robot moves, the extracted groups are tracked by extracting masks around each extracted group and correlating them under the constraints provided by translational flow lines (though the snake matching techniques could also be employed and simplified by the translational motion constraint). These constitute landmarks. Each landmark has static measures associated with it based upon the strength of the extracted group, it's distinctiveness with respect to surrounding portions of the image, and height relative to the horizon line defined by the immediate ground plane. There are also dynamic attribute which are updated as the robot moves for the number of frames in which the landmark stays in view, strength of frame-to-frame correlation, and the extent of cumulative image motion. A new viewframe is extracted when one of these landmarks disappears or a new one appears or the ordering of extracted landmarks changes.

3. REACQUISITION OF VIEWFRAMES

The robot acquires its current viewframe and posits that it has seen some subset of the landmarks previously. Initially this can be done by cueing the robot; as it proceeds, the sequencing of landmarks in memory limits the search space, as presented in [Levitt et al. - 1987a]. Given a maximum set of N possible landmarks the robot has viewed previously, there are $N!$ possible matches of the current viewframe to the previous. However, some landmarks in the current viewframe may not have been acquired previously. Therefore there are

$$\sum_{k=3}^{N} \binom{N}{k}^2 (k!)$$

possible matches of subsets of three or more landmarks in the current viewframe to those previously observed.

We can model each match as a hypothesis of correct association of landmarks. The size of this hypothesis space is forbidding even for six to ten landmarks, which is a typical number we might use in a viewframe. However, even with worst-case range estimates, given a single anchoring landmark match, only about 25% of the landmarks in the original viewframe panorama can be mis-matched. This assumes a roughly uniform 360° distribution of landmarks in viewframes (which is necessary for robust localization anyway). Figure 2 illustrates this situation. Here the square brackets on the radial lines indicate interval bounds of range estimates to a landmark.

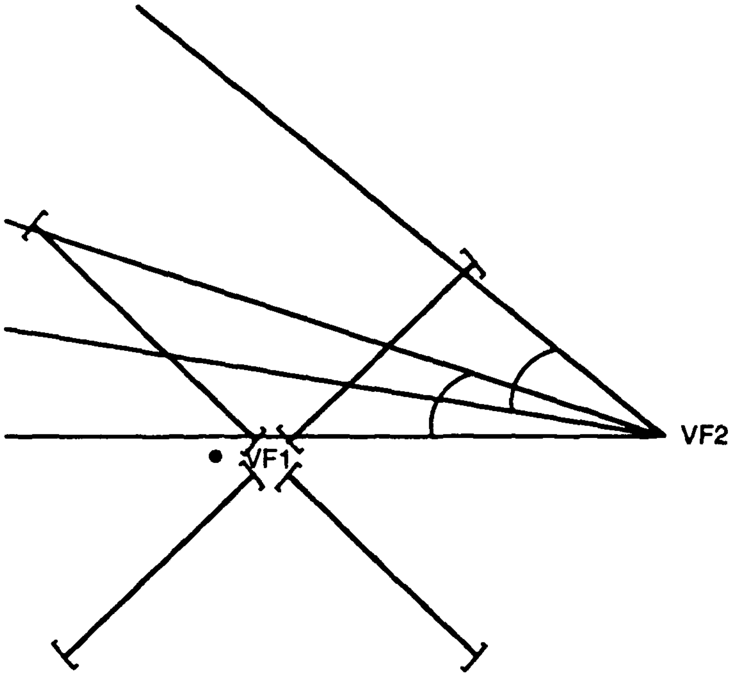


Figure 2: Overlap of Possible Matches

With this approximate limitation, the size of the hypothesis space is

$$\sum_{k=3}^{N} \binom{N}{\frac{k}{4}}^{2} \left(\frac{k}{4}!\right)$$

which for $N = 8$ is only 3328 hypotheses. Although the hypothesis space limitation does not scale up, the numbers of landmarks are typically low, and additional anchoring landmarks can further reduce the search space in practice.

Reacquisition begins with an initial match of at least three previously viewed landmarks. From these we form a common coordinate system between the viewframe in memory, VF1, and the currently perceived viewframe, VF2, as explained in [Levitt et al. - 1987a]. In this landmark-based coordinate system we can form regions about VF1 and VF2, called localizations. The vector between centroids is the best estimate of direction to move from VF2 to VF1.

The basic idea is to move toward VF1, observing the change in angles between landmarks, and updating range estimates and perceptual matches. This provides evidence that can be used to confirm or deny the multiple viewframe matching hypotheses. More specifically, the approach is to:

1. hypothesize the robot's location relative to the original viewframe by matching (subsets) of visual landmark descriptions
2. predict the location and occlusion behavior of landmarks after (small) motions of the robot

4. ISSUES

The key issue is robustness of the change in angle estimates between pairs of landmarks based on motion between viewframes. The angle estimates depend on the range estimates, which is the fundamental source of error. If one of the viewframe matching hypotheses is true, then even with poor range estimates, that hypothesis will converge with either the probabilistic or control theoretic approaches. However, we do not yet understand the sensitivity of motion along $\overrightarrow{VF}$ to convergence behavior.

The other major issue is perceptual stability of landmarks. Given typical landmarks as we define them, we expect anti-parallelism and relative (i.e. inter-landmark) size estimates to be stable, if we observe the landmarks under conditions of sufficient contrast. Color stability is, of course, problematic. Measures of anti-parallelism are also range dependent, based on the number of pixels on observed boundaries, and so cannot be depended on for great accuracy.

ACKNOWLEDGMENTS

This document was prepared by Advanced Decision Systems (ADS) of Mountain View, California, under U.S. Government contract number DACA76-85-C-0005 for the U.S. Army Engineer Topographic Laboratories (ETL), Fort Belvoir Virginia, and the Defense Advanced Research Projects Agency (DARPA), Arlington, Virginia. The authors wish to thank Sachi Toyofuku for providing administration, coordination, and document preparation support.

References

[Binford, Levitt, and Mann - 89] T. O. Binford, T.S. Levitt, and W. M. Mann, "Bayesian Inference in Machine Vision" in Uncertainty in AI III, T. S. Levitt, L. Konal and J. F. Lemmer (Eds.), North Holland, 1989 (in press).

[Kuipers - 82] B. J. Kuipers, "The 'Map in the Head' Metaphor," *Environment and Behavior*, vol. 14, no. 2, March 1982, pp. 202-220.

[Kuipers - 78] B. J. Kuipers, "Modeling Spatial Knowledge," *Cognitive Science*, vol. 2, 1978, pp. 129-153.

[Kuipers - 77] - B. Kuipers, "Representing Knowledge of Large-Scale Space", Massachusetts Institute of Technology, Artificial Intelligence Laboratory Technical Report, AI-TR-418, July 1977.

[Kuipers and Levitt - 88] B. J. Kuipers and T. S. Levitt, "Navigation and Mapping in Large Scale Space", *AI Magazine*, July, 1988.

[Lawton - 83] D. Lawton, "Processing Translational Motion Sequences", *Computer Vision, Graphics, and Image Processing*, vol. 22, 1983, pp. 116-144.

[Lawton, Levitt, and Gelband - 88] D. Lawton, T. Levitt, and P. Gelband, "Knowledge Based Vision for Terrestrial Robots", in *Proceedings of the DARPA Image Understanding Workshop*, Cambridge, MA, April 1988.

Levitt et al. - 88] T. Levitt, D. Lawton, D. Chelberg, K. Koitzsch, and J. Dye, "Qualitative Navigation II", in *Proceedings of the DARPA Image Understanding Workshop*, Cambridge, MA, April 1988.

Levitt et al. - 1987a] T. Levitt, D. Lawton, D. Chelberg and P. Nelson, "Qualitative Navigation", in *Proceedings of the DARPA Image Understanding Workshop*, Los Angeles, CA, February, 1987.

Levitt et al. - 1987b] T. Levitt, D. Lawton, D. Chelberg, P. Nelson, and J. Dye, "Visual Memory Structure for a Mobile Robot", in *Proceedings of the IEEE Workshop on Spatial Reasoning and Multisensor Fusion*, Morgan Kaufmann Publishers, Los Altos, CA, October, 1987.

Levitt et al. - 87] T. Levitt, D. Lawton, D. Chelberg, and P. Nelson, "Qualitative Landmark-Based Path Planning and Following", *AAAI-87 National Conference on Artificial Intelligence*, Seattle, WA, July 1987.

VERIFICATION OF RECOGNITION AND ALIGNMENT HYPOTHESES BY MEANS OF EDGE VERIFICATION STATISTICS [1]

A. J. Heller and J. R. Stenstrom

GE Corporate Research and Development Center
Schenectady, NY 12301

ABSTRACT

With complex scenes, most recognition systems produce alternate hypotheses regarding the location and orientation of objects present. We consider the most direct technique, identifying visible edges for a match hypothesis, and consider both the initial segmentation as well as underlying statistical support for the those edges that ought to be visible in the image.

INTRODUCTION

As model matching algorithms are applied to larger and larger images the probability of false matches being reported increases. In our work with the vertex-pair Hough-space-clustering-based matcher developed in our lab [Thompson and Mundy, 1987], we have found that when looking for instances of small objects in very large, highly cluttered images (greater than 1k x 1k pixels) that it is not uncommon to have the matcher propose as many as 50 alignments. This is due to the fact that in large images there are simply more features so that the probability of a the existence of a random grouping of features that aligns with the selected model features becomes significant.

In addition, we are now employing an automatic model feature selection algorithm [Mundy, et al., 1988], which typically chooses a larger number of features than a human operator would in order to guarantee a well-characterized model. Naively, we might try to tighten the cluster tolerances and demand a larger number of model-image correspondences to consider a given alignment correct.

There is considerable uncertainty about the extracted edges from the original image due to unpredictable and non-uniform illumination, self and multiple occlusion, sensor noise, texture, and possible camouflage. Most particularly, quantization limitations are problematic when considering smaller objects in larger images. Thus edges cannot be reliably found in every instance with any edge detection approach and as the objects occur across fewer pixels the edge localization and orientation becomes less exact. Moreover, without heuristics on edge strength and length the number of segmentation features becomes intractable and the accidental presence of false alignments becomes even more severe. Because of the limitations on the initial image segmentation, only a fraction of the chosen model features can be expected to be found at a given instance of the model in the image. Finally, since we are using a larger number of model features without being able to expect a proportionately larger number of corresponding image features, a greater number of false matches is expected.

These factors combine to make some form of hypothesis verification using other information available in the image or segmentation desirable.

[1] Work at GE was supported in part by the DARPA Strategic Computing Vision Program in conjunction with the Army Engineer Topographic Laboratories under Contract No. DACA76-86-C-0007 and the Air Force Office of Scientific Research under Contract No. F49620-89-C-0033.

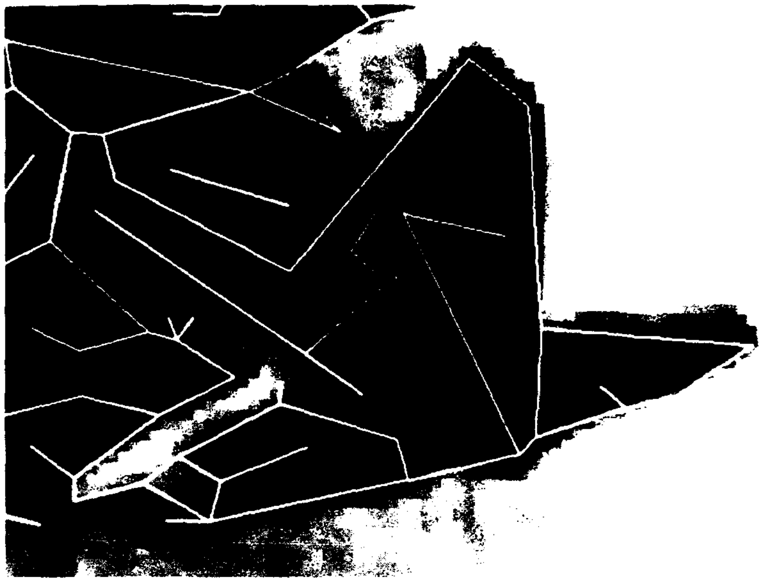


Figure 1: An image of the tail section of a C130 aircraft with segmentation edges overlayed showing the effects of camouflage.

Additionally, the "black magic" in the choice of cluster tolerance parameters is eliminated. The Hough-space search can be conducted with fairly wide tolerance parameters knowing that the verification stage will eliminate the ersatz alignments introduced by the more liberal parameter settings.

PRIOR WORK

Lowe's SCERPO Vision System [Lowe, 1985] has a verification module in which, among other things, potential matches between projected model features and segmentation features are evaluated according to the accuracy of agreement in position, length and orientation. Huttenlocher's ORA system [Huttenlocher, 1988] uses the fraction of the model contour which can be confirmed by edges in the segmentation as a criterion to verify alignments.

OUR WORK

In this report, we examine two methods for hypothesis verification. Each is based on the assumption that correct alignments should predict the location and orientation of features (notably edges) other than those used for the initial matching stage. Correct predictions are taken to be positive evidence for a corrpct alignment. The two methods differ in that the first, the simpler of the two, examines the segmentation for this evidence and the second looks for statistical support for edge hypotheses in the original image data.

In contrast with other approaches, we do not consider segmentation or image features that appear to conflict with the model feature being confirmed to be negative evidence for the correctness of a given hypothesis. In our experience, these conflicting features are exactly what is produced by military camouflage. Figure 1 is a detail of an image showing the tail section of a C130 aircraft with the segmentation edges overlayed to illustrate the conflicting edges introduced by camouflage.

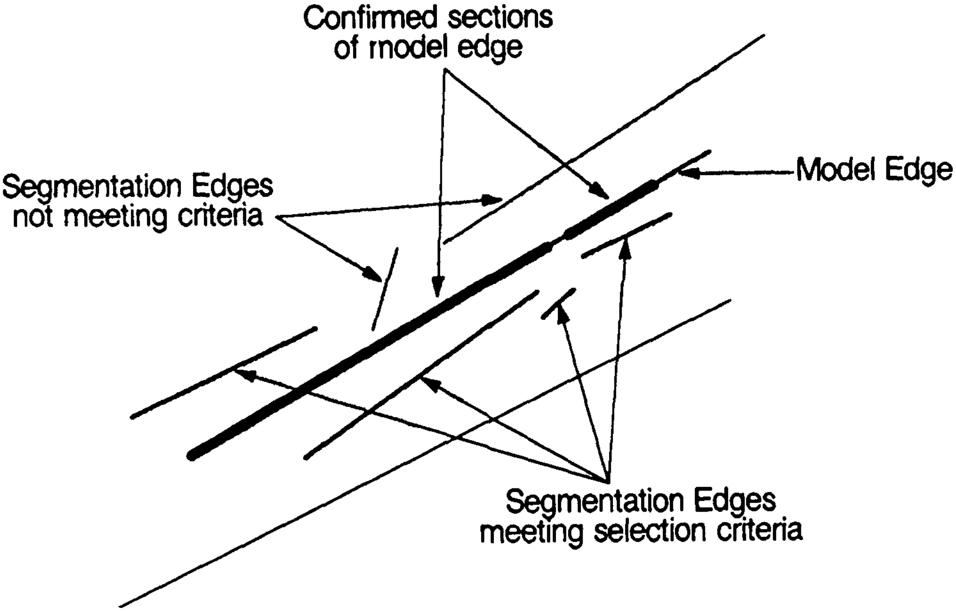

Confirmed sections of model edge
Segmentation Edges not meeting criteria
Model Edge
Segmentation Edges meeting selection criteria

Figure 2: A model edge confirmed by segmentation edges.

SEGMENTATION-BASED VERIFICATION

INTRODUCTION

The automatic model feature selection algorithm provides us with a well-characterized model in the sense that the model features chosen are visible and provide reliable and precise alignments over a wide range of viewpoints. This does not imply, that other model features, while not being able to provide as precise information about the initial alignment, do not provide useful evidence for the correctness of a given alignment. It is therefore useful to reexamine the segmentation to see how many additional features predicted by the model alignment are actually present.

DESCRIPTION OF ALGORITHM

This algorithm first eliminates edges from the model that are not visible in the proposed alignment using a simple back-face cull. Then the remaining edges are projected into the segmentation plane and those shorter that 10 pixels are discarded. Call this set E. For each edge e in E, we select the subset of edges in the segmentation whose endpoints are within 10 pixels and whose orientations are within 0.1 radians of e. Call this set S. These selected segmentation edges, the members of the set S, are then projected onto e, the model edge under consideration. The lengths of these projections onto e are summed, taking care to account for overlapping projections only once. This sum, the length of e accounted for by edges the edges in S, is divided by the length of e to obtain the confidence figure for the edge e, $P_l(e)$. In the current implementation, we require $P_l(e) > \frac{1}{2}$ to assert that the edge e has been confirmed by the segmentation.

An example of an edge confirmed is shown in Figure 2.

Finally, the ratio of the number of confirmed edges to the number of visible edges for a given model alignment is used as the confidence figure for that alignment:

$$P_{align} = \frac{|\{\forall e : e \in E \wedge P_l(e) > \frac{1}{2}\}|}{|E|}$$

IMAGE-BASED ALIGNMENT VERIFICATION

INTRODUCTION

For a given image, there is no ideal segmentation into edges and vertices. All segmentation algorithms incorporate compromises. For example, our matching system relies on having accurate angles between edges in the segmentation, therefore long edges in the segmentation are desirable; but if one gets overly zealous about demanding long edges, the direction of the edges can become unreliable because the edges no longer follow the image contours accurately.

The requirement that a segmentation method be useful for a wide variety of images suggests that edges may exist in a given image which are not found because of the necessarily general nature of the particular approach. Let us, then, first consider the question of exactly what characterizes an edge in an intensity image.

It is the essence of edge-based image understanding to define the locations of change from one image property to another. Most often this reduces simply to defining an edge of occlusion of one surface by another. With classic lighting and reflectance models this leads to looking for suitably high gradients of the intensity field and attempting to optimize the position and direction of the edge.

This type of continuous analysis has two problems:

- The intensity function is expected to be discontinuous at points of interest, so smoothing and curve fitting in the neighborhood becomes necessary to make a discrete physical phenomenon fit into a simple continuous and differentiable model.
- The plan, while often quite successful, responds only to changes between the average intensities of regions. Typically other changes in the statistical distribution of pixels around an edge are lost.

A STATISTICAL CLASSIFIER APPROACH TO EDGE VERIFICATION

Consider a relatively basic definition for an edge pixel and edge direction. An edge pixel whose direction is d separates those pixels along direction d into two samples, one to the left and one to the right, where there is statistical support that these two samples were not drawn from the same population.

If we consider the noise to be additive and distributed according to the Gaussian, then within regions of constant illumination, reflectance, and apparent angle, the image intensity represents samples of a normal random variable. The most common case is adjacent polygons of Lambertian reflectance. This suggests using a parametric test which compares the overlap of the two distributions, such as ***Student's t-test*** or the ***Snedecor F-test*** on the possible equivalence of two populations giving rise to the samples at a candidate edge pixel. At least one edge extractor uses such a statistical approach [Kundu and Mitra, 1987].

In our application, however, we are testing whether or not an edge is present in the image in a certain region near the projected model edge. Consider the case where the candidate edge for confirmation is slightly in error and therefore does not create a proper division of the samples at the location, but two distinct populations do exist in the neighborhood – there is an edge nearby but not at the location and direction under consideration. Near a true edge, the assumption that samples are being drawn from two distinct normal random variables, one fully on each side of the edge represented by the pixel and direction is incorrect more often than not.

It is possible to to perform tests on the pixel divisions, insuring that they may be expected to have been drawn from a normal random variable. However, sample sizes will be tiny for this purpose and there will be little chance to reject the hypothesis.

In practice, we choose to work with a different test that does not have the same requirements on the underlying sample distributions. The *chi-square test* is a simple test to determine the probability of chance variation between expected and observed frequency distributions.

Where e_i is the expected count in the i^{th} category and o_i is the observed count we have:

$$\chi^2 = \sum_{i=0}^{N} \frac{(o_i - e_i)^2}{e_i}$$

Letting the degrees of freedom, $\nu = N - c - 1$, where N is the total number of categories and c is the number of constraints, the probability $P_\chi(\chi^2, \nu)$ that any random set of data points with ν degrees of freedom would yield a chi-square as large or larger than χ^2 is:

$$P_\chi(\chi^2, \nu) = \int_{\chi^2}^{\infty} \frac{z^{(\nu-2)/2} e^{-x/2}}{2^{\nu/2}\Gamma(\nu/2)} dz = e^{-\chi^2/2} \sum_{m=0}^{(\nu-2)/2} \frac{(\chi^2/2)^m}{m!} \quad \nu \text{ even}$$

This is a well-known statistic and the reader is referred to [Alexander, 1961], [Bevington, 1969], and [Press, et al., 1986] for a full discussion of its derivation, application, and implementation. In cases where parametric tests are applicable, more information is known and the chi-square test is a weaker test. If there is the presumption of applicability, there will be cases where two normal random variables are different slightly in mean or variance and the t-test or F-test will prove significant but a chi-square test will not. Nevertheless, we will see that the chi-square test does quite well in the current context. Future work will consider the utility of making this presumption and considering the consequent edges.

It should be noted too, that the choice of the chi-square test relaxes the requirement that the random variables being sampled be normally distributed. This may be important when there is texture, the lighting model is more complex, the noise in the image is not Gaussian, or there are more than two populations near an edge pixel.

In practice, we work with square neighborhoods in the vicinity of candidate edge pixels, just as with a typical edge detector. We simply apply statistical decision procedures to the discrete data present rather than casting the data into continuous functions.

DESCRIPTION OF THE ALGORITHM

In this algorithm we go through the same first steps as in the segmentation-based verification to obtain the set of edges we are attempting to confirm – a simple edge visibility check, projection of edges into the image plane and elimination of short edges. For each of these remaining projected edges we consider the pixels in the two regions on either side of the edge. (The pixels lying directly on the edge and those within one pixel to either side are ignored to allow for small errors in the alignment of the edge.)

The two groups of pixels are examined with the chi-square test. We are testing the null-hypothesis that there is no edge in this neighborhood. If the edge passes through completely homogeneous area of the image,

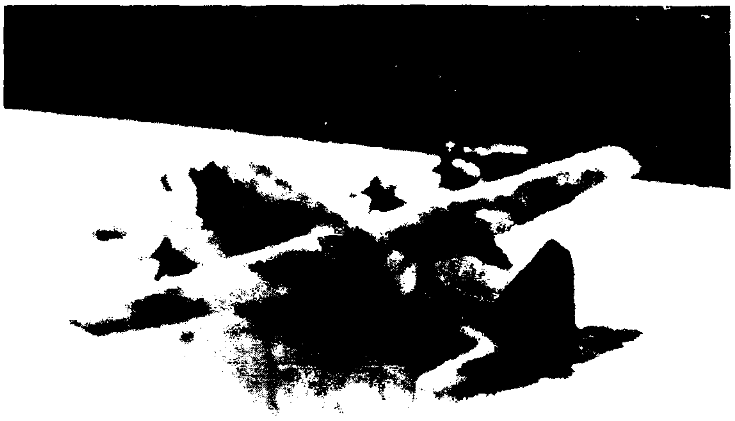


Figure 3: A small section of an image of a nearby Air National Guard Base.

we find that $P_\chi \approx 1$, thus confirming the null hypothesis. If the edge falls along a very well-defined pixel boundary in the image, then $P_\chi \approx 0$. Intermediate values of P_χ indicate that we are uncertain about the presence of the given edge, with smaller values corresponding to greater certainty about the edge's existence. In this implementation, we require that $P_\chi < \frac{1}{10}$ (i.e. we are 90 percent certain that the pixels from the two sides of the edge are from different distributions) to assert the existence of the edge under test in the image. As in the previous algorithm, the ratio of the number of confirmed edges to the number of edges visible at that alignment is used as the confidence figure, P_{align}, for the proposed alignment. That is, if E is the set of projected edges that are greater than 10 pixels long and whose corresponding model edges are visible, then

$$P_{align} = \frac{|\{\forall e : e \in E \wedge P_\chi(e) < \frac{1}{10}\}|}{|E|}$$

EXPERIMENTAL RESULTS AND DISCUSSION

To demonstrate these methods in a convenient manner we used the 200 x 150 pixel subsection shown in Figure 3 of an image of an Air National Guard Base (The 109th Tactical Airlift Group, located near the GE Research and Development Center in Schenectady, NY). A prominent feature of this subsection of the image is a C130 transport aircraft. As one can see by examining the corresponding segmentation, shown in Figure 4, the camouflage on the aircraft and the cracks in the concrete add a considerable amount of clutter to this scene and corrupt many of the longer edges in the image.

Figure 5 shows the simple face-edge-vertex model of the C130 aircraft used in these experiments.

To simulate the situation in a very large image, where many possible alignments would be proposed by the vertex-pair matcher, the cluster tolerance parameters for this match were set to values much larger than would normally be indicated for an image of this size. Figure 6 shows the resultant model alignments overlayed on the original image. We should emphasize that under normal circumstances when working with

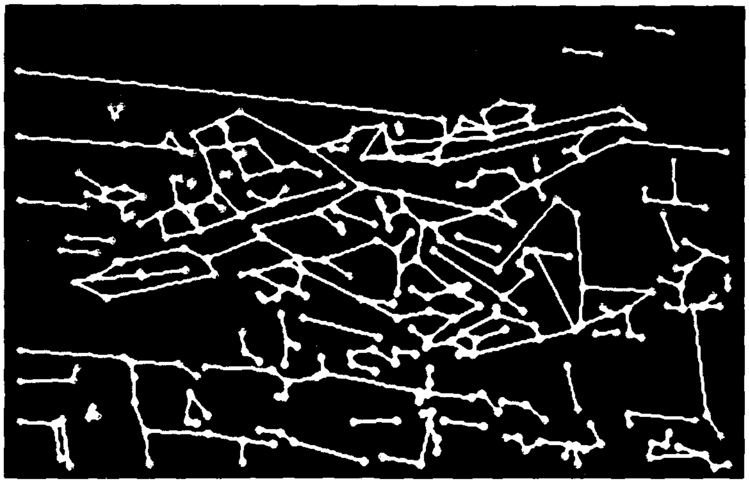


Figure 4: A segmentation of the image in the previous figure.

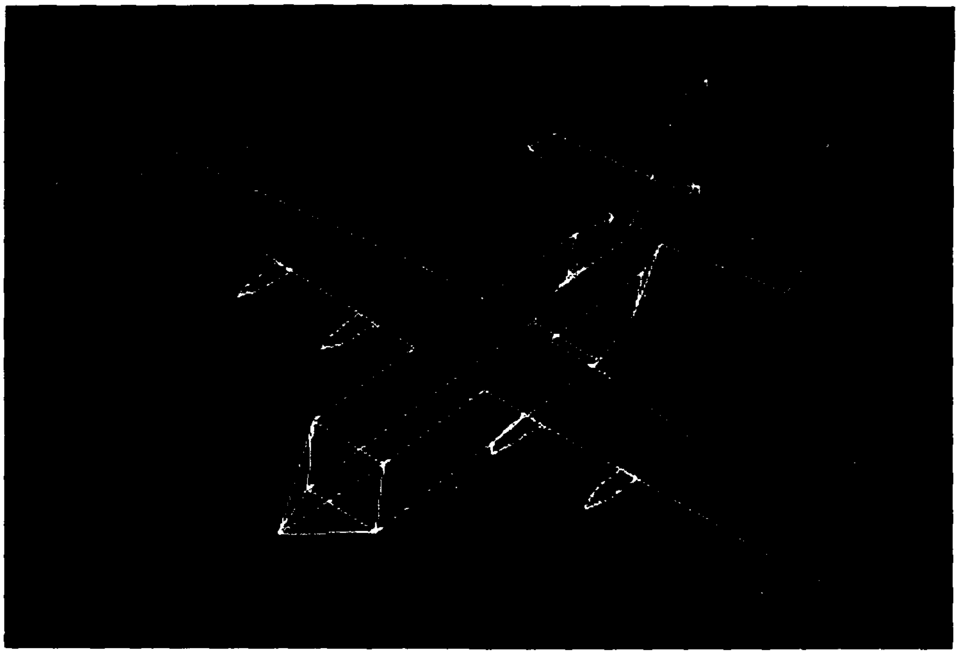


Figure 5: A simple face-edge-vertex model of a C130 aircraft.

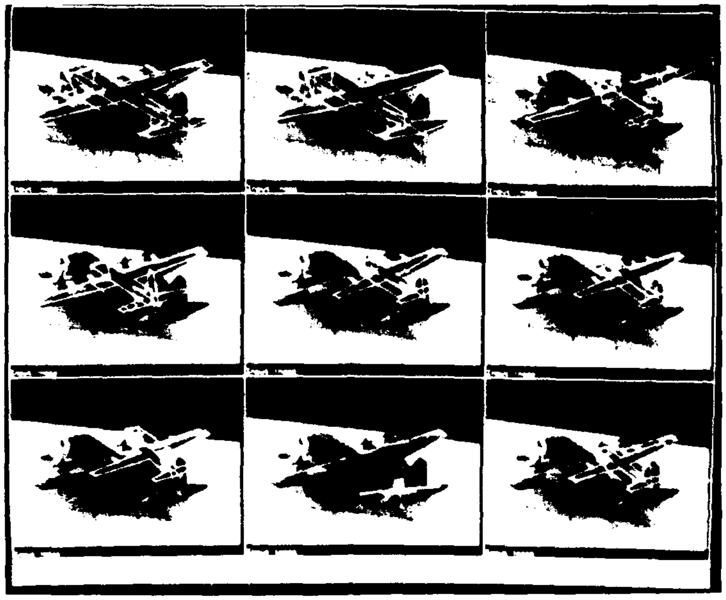


Figure 6: Some proposed model alignments.

images of this size, only the left and center alignments in the top row would be proposed by the matcher. Each of these alignments were evaluated by both approaches outlined in this report and ranked by the resultant confidence figures.

The alignment assigned the highest confidence figure ($P_{align} = 0.66$) by the segmentation-based verifier is shown in Figure 7. Figure 8 show the alignment assigned the highest confidence figure ($P_{align} = 0.73$) by the image-based verifier. Each approach chose as the second ranked alignment the top-ranked one of the other approach. The remaining alignments shown in Figure 6 were ranked considerably worse ($P_{align} < 0.40$) by both approaches. It has been our experience that confidence figures greater than $\frac{1}{2}$ (i.e. $P_{align} > \frac{1}{2}$) appear to correspond to what we would consider "good" matches. Work is currently underway to derive this criterion in a formal manner, by analyzing the distributions of the confidence figures.

We also observe that the top-ranking alignment determined by the statistical qualifier approach aligns with the C130 image better than the one determined by the segmentation-based approach. This confirms our experience that for verification purposes the chi-square test is quite robust, typically yielding negligible values when there is no edge upon human inspection and large values when there is an edge apparent.

This result, along with a scheme to establish equivalence classes of alignments, would form the basis of a system that could reliably distinguish multiple instances of the model in an image from multiple false matches. This is a frequently encountered situation in our work with images.

Another area of investigation suggested by the success of the image-based statistical classifier approach is to exploit face information contained in the model. Just as a valid projection of a model edge divides the pixels to either side of it into statistically distinct populations, a valid projection of a model face should divide the enclosed pixels into distinct populations. This face-information-based verification method would provide a higher level of coherence to the image topology. If face-based verification could be demonstrated, it would provide strong evidence for the validity of a given alignment.

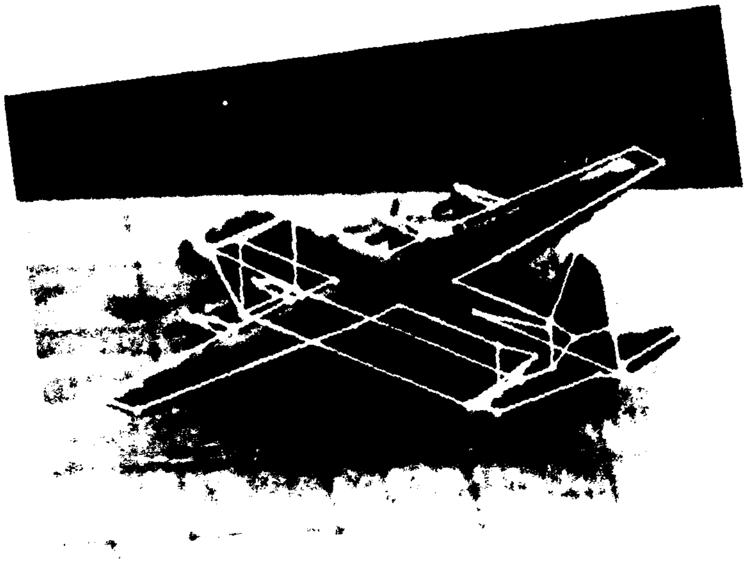


Figure 7: The best match as determined by the segmentation based verifier.

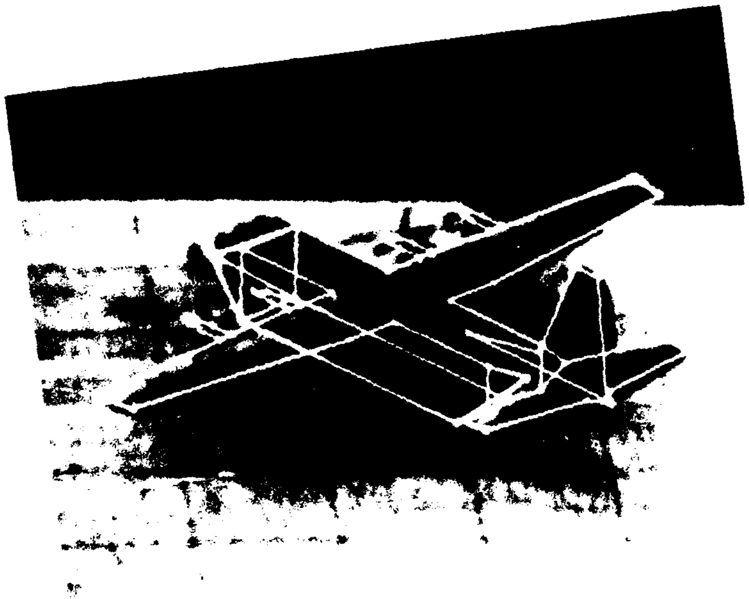


Figure 8: The best match as determined by the image based verifier.

References

[Thompson and Mundy, 1987] Thompson, D. W., and J. L. Mundy, "Three-Dimensional Model Matching From an Unconstrained Viewpoint," *Proc. IEEE Robotics and Automation*, 1987, pp. 280.

[Mundy, et al., 1988] Mundy, J. L., A. J. Heller, and D. W. Thompson, "The Concept of an Effective Viewpoint," *Proc. DARPA Image Understanding Workshop*, April 1988, vol. 2, pp. 651-659.

[Lowe, 1985] Lowe, D., *Perceptual Organization and Visual Recognition*. Boston: Kluwer Academic Publishers, 1985.

[Huttenlocher, 1988] Huttenlocher, D. P., "Three-Dimensional Recognition of Solid Objects from a Two-Dimensional Image," Ph.D. Thesis, MIT Dept. of Elect. Eng. and Comp. Sci., 1988.

[Kundu and Mitra, 1987] Kundu, A., and S. K. Mitra, "A New Algorithm for Image Edge Extraction Using a Statistical Classifier Approach," *IEEE Trans. Pattern Anal. and Machine Intell.*, vol. PAMI-9, pp. 569-577, July 1987.

[Alexander, 1961] Alexander, H. W., *Elements of Mathematical Statistics*. New York: John Wiley and Sons, 1961.

[Bevington, 1969] Bevington, P. R., *Data Reduction and Error Analysis for the Physical Sciences*. New York: McGraw-Hill, 1969

[Press, et al., 1986] Press, W. H., et al., *Numerical Recipes, The Art of Scientific Computing*. Cambridge: Cambridge University Press, 1986.

Straight Homogeneous Generalized Cylinders: Analysis of Reflectance Properties and a Necessary Condition for Class Membership*

Ari D. Gross and Terrance E. Boult

Columbia University Department of Computer Science
New York City, New York, 10027 ari@cs.columbia.edu, tboult@cs.columbia.edu

Abstract

Generalized cylinders have been the focus of considerable vision research. One class of generalized cylinders that has been studied is that of straight homogeneous generalized cylinders, whose cross sections are scaled versions of a reference curve. Much of the research into generalized cylinders first *assumes* that an object in the image is that of a generalized cylinder and then analyzes image features of the object, such as contour, in order to recover properties of the underlying shape. In this paper, we consider a membership test for straight homogeneous generalized cylinders to determine *if* the object in the image is a member of the shape class. First, some reflectance constraints for the straight homogeneous generalized cylinders are developed. Using these reflectance constraints, we derive a necessary condition to test images for membership in the straight homogeneous generalized cylinder shape class. This test is restrictive in that it only applies to a straight homogeneous generalized cylinder in *semi-canonical* position. A more general membership test is derived for linear straight homogeneous generalized cylinders. Examples of the performance of these membership tests on simulated, noisy intensity images are provided.

1 Introduction

A generalized cylinder is a solid defined by its axis, cross-section, and sweeping rule. The importance of generalized cylinders (hereafter GC's) as a representation is that they are *general* enough to represent many real-world objects yet sufficiently well-defined (at least for certain subclasses) that we can attempt to recover shape parameters from image data. They have been the topic of considerable research in computer vision [Brok81],[Mar77],[Pon88],[Shaf85]. Because GC's are a very expressive representation, however, recovery of shape parameters from image data has proven to be difficult.

Marr proposed [Mar77] that shape, to a large degree, can be inferred from image contour. Shafer [Shaf85] creates a GC taxonomy and studies several of these GC classes in detail; he proves several important theorems about GC's. In particular, Shafer considered a subclass of GC's known as straight homogeneous generalized cylinders (hereafter SHGC's). SHGC's are defined as a proper subclass of GC's generated by scaling a reference cross-section curve along a straight axis. More recently, Ponce [Pon88],[PCM87] studied the projective properties of GC's, generalized some of the results found in Shafer, and proved certain uniqueness and shape from contour results for SHGC's. Other recent work on contour analysis involves finding certain symmetries in the image contour and using these cues to infer properties of the underlying 3D shape [Nal87],[UN88],[RM88].

In this paper, we use both contour *and* intensity information to derive a membership test for SHGC's. The purpose of the membership test is to determine if a part of the image can be the projection of an underlying SHGC solid. Such a test would be particularly valuable in a polymorphic shape recovery system. A polymorphic shape recovery system, able to recognize many different classes of shape, would benefit from the ability to determine whether an image section is part of an SHGC *before* starting to actually fit an SHGC to the image data.

There is a subclass of SHGC's that, it turns out, are particularly amenable to a membership test. This class, linear straight homogeneous generalized cylinders (hereafter LSHGC's), is a proper subclass of SHGC's where the sweeping function is restricted to be linear. Although SHGC's as a whole are considered in this paper, LSHGC's will be specifically examined. The membership tests that are derived in this paper are necessary, not sufficient, conditions for class membership.

2 Finding Reflectance Constraints Along Image Meridians

In the sequel, we assume that the SHGC surfaces have lambertian reflectance properties. Orthographic projection is assumed throughout the paper. We have adopted the parameterization for SHGC's from Ponce [PCM87], including a requirement that the cross-section function be star-shaped.

* Support for this work was provided in part by the Advanced Research Projects Agency of the Department of Defense under contract #N00039-84-C-0165 and by the National Science Foundation.

An SHGC, restricted to star-shaped cross-section, with axis aligned along $\vec{k}$ can be written as

$$\vec{OP}(z, \theta) = f(z)\, g(\theta)\, (\cos\theta\vec{i} + \sin\theta\vec{j}) + z\vec{k}$$

The function f(z) is the sweeping rule of the GC, which associates with each z value a scaling factor of the cross-section curve. The function g(θ) is the cross-section curve, which varies only in scale for different values of z.

The surface normal at a point $\vec{OP}(z, \theta)$ on the SHGC surface is given by the vector product of the partial derivatives of $\vec{OP}$ with respect to θ and z. The partial derivatives of $\vec{OP}$ are given by

$$\frac{\partial\vec{OP}}{\partial\theta} = f(z)\,(g'(\theta)\cos(\theta) - g(\theta)\sin(\theta))\,\vec{i} + f(\cdot)\,(g'(\theta)\sin(\theta) + g(\theta)\cos(\theta))\,\vec{j}$$

$$\frac{\partial\vec{OP}}{\partial z} = f'(z)\, g(\theta)\cos(\theta)\,\vec{i} + f'(z)\, g(\theta)\sin(\theta)\,\vec{j} + \vec{k}$$

The normal $\vec{n}$ is just the cross product of these partials derivatives and is given by (for conciseness z and θ are generally omitted hereafter)

$$\vec{n} = (g' f \sin(\theta) + g f \cos(\theta))\,\vec{i} + (-g' f \cos(\theta) + g f \sin(\theta))\,\vec{j} + g^2 f f' \tag{2.1}$$

The cross-section curves at every point along the SHGC axis are the same, except for scaling, making them parallel curves to one other [DoC76]. A meridian of the SHGC is a curve along the surface of the SHGC that intersects every parallel curve (i.e., cross-section curve) at the same value of θ. In Figure 1 an SHGC is shown (in canonical position with respect to the viewer) with some of its parallel curves and meridian lines displayed. We are interested in whether there is some correlation of image brightness values among points lying on the same meridian of an SHGC surface. If such a correlation exists, we then need to find a procedure (if one exists) to find *image meridians*, which are defined as lines in the image plane (not necessarily straight) that are projections of meridians on the surface of the SHGC.

To determine the image brightness for points lying on the same image meridian, we need to assume some initial imaging model. Let us assume that there is a single distant point light source in the $-\vec{j}$ direction. Let us further assume that the viewing direction is aligned with the light source direction and is also in the $-\vec{j}$ direction.

To use the standard image brightness equations, we need the p and q values for each point $\vec{OP}(z, \theta)$ on the SHGC. We want a brightness equation to map each set of (z, θ) values, corresponding to a point on the SHGC surface, into some value equal to the perceived brightness of that point in the image plane. The reflectance function R(p, q) [BB82] maps (p, q) values into image intensities, where p and q are the surface gradients. The reflectance function when viewer and light source positions are aligned is given by

$$R(p, q) = \frac{r_0}{\sqrt{1 + p^2 + q^2}}$$

Since we are concerned with intensity correlation among points on an image meridian, we prefer a reflectance function that maps (z, θ) values to image intensities. The surface gradients in our example, with $-\vec{j}$ the viewing direction and the imaging plane parallel to the $(\vec{i}, \vec{k})$ plane, are obtained by dividing the $\vec{i}$ and $\vec{k}$ components of the surface normal given in Equation (2.1) by the $\vec{j}$ component. From Equation (2.1) we have

$$p(z, \theta) = \frac{g'(\theta) f(z) \sin(\theta) + g(\theta) f(z) \cos(\theta)}{g'(\theta) f(z) \cos(\theta) - g(\theta) f(z) \sin(\theta)}, \quad q(z, \theta) = \frac{g(\theta)^2 f(z) f'(z)}{g'(\theta) f(z) \cos(\theta) - g(\theta) f(z) \sin(\theta)}$$

The reflectance function R(z, θ) provides the image brightness of a point $\vec{OP}(z, \theta)$ on the SHGC surface, and is given by

$$R(z, \theta) = \frac{r_0}{\sqrt{1 + p^2(z, \theta) + q^2(z, \theta)}} \tag{2.2}$$

where p(z, θ) and q(z, θ) are as defined above.

Consider two points $P_1 = \vec{OP}(z_1, \theta_0)$ and $P_2 = \vec{OP}(z_2, \theta_0)$ that lie on the same meridian of an SHGC surface. Using Equation (2.2) we obtain

$$\frac{1}{R(z_1, \theta_0)^2} - \frac{1}{R(z_2, \theta_0)^2} = (f'^2(z_1) - f'^2(z_2)) \cdot \frac{g^4}{r_0^2 (g' \cos(\theta_0) - g \sin(\theta_0))^2} \tag{2.3}$$

We want a constraint among image brightness values of points lying along an SHGC meridian that is independent of θ. To do this, we take yet another point $P_3 = \vec{OP}(z_3, \theta_0)$, lying along the same surface meridian as P_1 and P_2. Using the three meridian points P_1, P_2, and P_3 and Equation (2.3), we derive the following constraint

$$\frac{\dfrac{1}{R_1^2}-\dfrac{1}{R_2^2}}{\dfrac{1}{R_1^2}-\dfrac{1}{R_3^2}} = \frac{f'^2(z_1)-f'^2(z_2)}{f'^2(z_1)-f'^2(z_3)} \tag{2.4}$$

where R_i is defined as $R(z_i, \theta_0)$.

Equation (2.4) provides a ratio among three meridian points P_1, P_2, and P_3 that is independent of θ and the cross-section curve $g(\theta)$. This ratio is strictly a function of z. If there were a procedure for finding image meridian points directly from the image, then Equation (2.4) would provide a necessary condition for an SHGC image, namely, that corresponding meridian points from three z values along the SHGC spine yield a constant ratio. A procedure for finding image meridians for SHGC's in *semi-canonical* viewing position and for LSHGC's in arbitrary viewing position is given in section 4.

3 Relaxing the Light Source Condition

The analysis of the previous section and the corresponding constraint provided by Equation (2.4) are restricted by the fact that the imaging model assumed that the light source and the viewer were both aligned in the same direction $(-\vec{j})$ with respect to the SHGC. This assumption is, of course, very restrictive. We seek to relax these constraints one at a time while still obtaining some reflectance constraints in the image that can be used to test for membership. In this section, we relax the restriction on light source position.

For a more general light position, where the light source is in the $(p_s, -1, q_s)$ direction, the radiance equation for a point $\vec{OP}(z, \theta)$ is given by

$$R(z,\theta) = \frac{r_0\,(p_s \cdot p + q_s \cdot q + 1)}{\sqrt{(1+p^2+q^2)\cdot(1+p_s^2+q_s^2)}} \tag{3.1}$$

where p and q are as defined in section 2.

For the case of an LSHGC, the analysis is simple. Both p and q are functions of θ alone (refer back to section 2), since $f(z)$ is constant and $f'(z) = 0$. As a result, iso-intensity contours will lie along image meridians since the value of $R(\cdot)$ in Equation (3.1) is also determined by θ alone. *So for the case of* LSHGC's, there is an image-observable constraint, iso-intensity contours along image meridians, that holds for arbitrary light source position.

For SHGC's, we examine the cases of $p_s \neq 0$ and $q_s \neq 0$ separately. In the case of $p_s \neq 0$, the image brightness at a point $\vec{OP}(z,\theta)$ is given by

$$R(z,\theta) = \frac{r_0\,(p_s \cdot p + 1)}{\sqrt{(1+p^2+q^2)\,(1+p_s^2)}}$$

where p and q are as defined in section 2. This equation can be manipulated exactly the same way as Equation (2.2), and yields a constraining reflectance equation identical to Equation (2.4). When $q_s \neq 0$ the reflectance equation is given by

$$R(z,\theta) = \frac{r_0\,(q_s \cdot q + 1)}{\sqrt{(1+p^2+q^2)\,(1+q_s^{2)}}}$$

where p and q are again as defined in section 2. This equation contains f' terms both in the numerator and denominator. Consequently, we cannot manipulate this reflectance equation as we did in the previous case. This equation does not seem to lend itself to a simple constraining reflectance equation .[1]

With regard to relaxing the light source condition (that in section 2 required that $p_s = q_s = 0$), we can sum up as follows: the case of LSHGC's present no problem; in the general case of SHGC's, the reflectance constraints apply only when $p_s \neq 0$, but not when $q_s \neq 0$.

4 Finding Image Meridians

In order to test an image to see if corresponding image meridian points for three values of z yield a constant ratio, as was derived in Equation (2.4), we need to be able to find image meridians directly from the image of an SHGC. We assume once again that the SHGC is aligned canonically with the viewer reference frame, i.e., the viewing direction is $-\vec{j}$. Contours in the image are projections of two kinds of contour generators, limbs and edges. Limbs of the SHGC, which concern us here, are *contour generators* where the surface turns away from the viewer. A limb point, then, has the property that $\vec{v} \cdot \vec{n} = 0$, where $\vec{v}$ is the viewing direction. In our case, the viewing plane is parallel to the $(\vec{i}, \vec{k})$ plane. Since $\vec{v}$ has only a

[1] What is meant by a *simple* constraining reflectance equation is one that is independent of solving for the $f(\cdot)$ and $g(\cdot)$ functions of the underlying SHGC.

single nonzero component of -1 in the direction of $\vec{j}$, the limb equation for SHGC's in canonical position is exactly where $\vec{n}$ has a j component of 0, so that the limb equation is given by

$$-g' \cos(\theta) + g \sin(\theta) = 0 \tag{4.1}$$

But Equation (4.1) is strictly a function of θ. Therefore, if θ_0 satisfies Equation (4.1) and z_i is a value of z lying within the SHGC image, then $\vec{OP}(z_i, \theta_0)$ is a point on the surface of the SHGC that projects onto the 2D contour of the SHGC in the viewer's image plane. Since the SHGC image axis in this canonical setting is in the $\vec{k}$ direction, the occluding contour to the right of the image axis results from a meridian on the SHGC surface with $\theta = \theta_0$, while the occluding contour to the left of the image axis results from a meridian of the SHGC surface with $\theta = \theta_1$.

Using the limb equation for canonically positioned SHGC's given by Equation (4.1), we can align points on the image of the SHGC that lie along an image meridian. To align meridian points, we first run one of the algorithms given in Ponce [PCM87] in order to obtain the SHGC image axis. The recovered axis is used to place a 2D coordinate system on the imaging plane. The recovered axis is the coordinate axis in the $\vec{k}$ direction. A coordinate axis in the $\vec{i}$ direction is placed at some height of the image plane orthogonal to the SHGC axis.

Consider a point P_1 at location $(x_1\vec{i}, z_1\vec{k})$ of the viewer's image plane (assuming the viewing direction is $-\vec{j}$) that is is the projection of a point $(x_1\vec{i}, y_1\vec{j}, z_1\vec{k})$ on the surface of the SHGC. For a given value of z, say z_2, we want to find a corresponding value of $x = x_2$ such that $P_2 = (x_2, z_2)$ is the projection onto the image plane of a point $(x_2\vec{i}, y_2\vec{j}, z_2\vec{k})$ that lies on the same surface meridian of the SHGC as $(x_1\vec{i}, y_1\vec{j}, z_1\vec{k})$. To find x_2 we first need to find the value of x_{c1} for $z = z_1$, where (x_{c1}, z_1) is a point on the SHGC image contour. Next, we find the corresponding point x_{c2} for $z = z_2$. Since the SHGC is aligned with the $(\vec{i}, \vec{k})$ image plane, $x_{c1} = f(z_1) \cdot g(\theta_0) \cdot \cos(\theta_0)$ and $x_{c2} = f(z_2) \cdot g(\theta_0) \cdot \cos(\theta_0)$, for some value $\theta = \theta_0$. Since we know the value of x_{c1} and x_{c2}, the ratio of

$$r = \frac{x_{c1}}{x_{c2}} = \frac{f(z_1) \cdot g(\theta_0) \cdot \cos(\theta_0)}{f(z_2) \cdot g(\theta_0) \cdot \cos(\theta_0)} = \frac{f(z_1)}{f(z_2)}$$

can be computed. The desired value for x_2 corresponding to x_1 is simply $\frac{x_1}{r}$ since

$$x_2 = \frac{x_1}{r} = \frac{f(z_1) \cdot g(\theta_1) \cdot \cos(\theta_1)}{r}$$

5.1 Relaxing the Viewing Position Condition

We now want to generalize the results of the previous section. Section 4 assumed that the SHGC was in canonical position vis a vis the viewer by assuming a viewing direction of $-\vec{j}$. Aligning image meridian points made use of assumptions *only* valid for an SHGC in canonical position. We would like to generalize this as much as possible. Can we determine points in the image that lie on the same image meridian when the SHGC is in an arbitrary viewing position?

Consider first a canonically positioned SHGC. What if the SHGC were rotated around the k axis (hereafter referred to as a k-rotation) by ϕ, as shown in Figure 2? Obviously, this rotated SHGC can be reparameterized as a canonical SHGC except that $g(\theta)$ is replaced by some new polar function $h(\theta)$. Since this SHGC *is* still in canonical position under some reparameterization of the underlying SHGC, image meridian points can still be determined from the image (as in the canonical case).

Now consider an SHGC initially in canonical position that is rotated by ψ around the j axis, as shown in Figure 3. For this type of rotation, which we refer to as a j-rotation, the intensity image of the SHGC is merely rotated in the viewer image plane, parallel to the $(\vec{i}, \vec{k}$ plane), by ψ. To restore this image to the projection of a canonically aligned SHGC, the algorithms given in Ponce [PCM87] for finding the 2D axis of an SHGC from its contour can be used. Once the image axis has been found, the image of the SHGC can be rotated in the image plane until the SHGC axis is aligned with $\vec{k}$. We now have a *restored* canonical image of an SHGC for which the method given in section 4 for finding image meridians can now be applied. An SHGC that was originally in canonical position and was then rotated about the j and k axes is said to be in *semi-canonical* position. Thus, an SHGC in semi-canonical position has a contour generator lying on meridians of the surface and its image meridians can be determined using the method described section 4.

We now consider rotating an SHGC from its canonical position around the i axis (towards or away from the viewer's line of sight), see Figure 4. The question that needs to be answered is: *does the occluding contour still lie on the meridian of an SHGC so that image meridian points can be detected in the image*? The limb equation for an SHGC that was rotated around the i axis is given by

$$\cos(\psi) \ (g \sin(\theta) - g' \cos(\theta)) = \sin(\theta) f' g^2 \tag{5.1}$$

Equation (5.1) is not simply a function of θ. It has an f' term, where f' is a function of z. With the exception of LSHGC's, where f' is constant, the contour generator of this rotated SHGC does *not* lie on the meridian of the underlying SHGC. Thus, the procedure given earlier in this paper for aligning meridian points in the image will fail.

For a general rotation of an SHGC in space, then, we have an alignment problem. Alignment of image meridian points is difficult since the occluding contour cannot be used to align the points as it is not guaranteed to be the projection of an SHGC meridian. This alignment problem is only true of SHGC's in general; for LSHGC's, the limb equation given in Equation (5.1) lies along a surface meridian.

5.2 Linear Straight Homogeneous Generalized Cylinders

In section 3 we showed that the reflectance constraints of section 2 for image meridians apply to an LSHGC with the light source in arbitrary position. It is also clear from section 5.1 that contour generators for LSHGC's in arbitrary viewing condition lie along surface meridians. It is possible then, based on section 4, to find image meridians. A membership test looks for points along the same image meridian to have the same intensity values. If the image appears to satisfy this constraint, it accepts it as a possible LSHGC image. It can then proceed to recover actual parameters of the underlying LSHGC. If the image does not meet the criterion, i.e., iso-intensity image meridians, it can be removed from further consideration as a possible LSHGC image since it failed a necessary test of class membership.

6 Experimental Results

In this section, we test how well the reflectance constraints of Equation (2.4) and Equation (3.1) perform when used as a test of class membership for SHGC's in semi-canonical position and LSHGC's in arbitrary position respectively. The images used were noisy simulated intensity images. Any intensity image generated from an underlying SHGC solid in semi-canonical position or from an LSHGC in arbitrary position, under the set of assumptions mentioned in s:ctions 3, 5.1, and 5.2, should satisfy the appropriate reflectance constraint. Satisfying the appropriate reflectance constraint, :n either the SHGC or LSHGC case, is a necessary but not sufficient condition that the underlying surface belongs to that cl:.ss of GC's. We test several positive cases to make sure the constraints are satisfied. In addition, a negative example is presented. Each image tested contained only one object, situated against a black background (greylevel = 0).

The test for LSHGC images is whether image meridians are of approximately constant intensity since, based on Equation (3.1), the reflectance function R is strictly a function of θ. The algorithm determines the intersection of the two sides of the image of the object. We extend a line from this point of intersection down the image plane in a direction parallel to the z axis. The horizontal lines are then aligned with each other, as described in section 4. Several image meridians are then tested to see if they satisfy the reflectance constraint of Equation (3.1); for LSHGC's points along the image meridian should have approximately the same intensity values. The points along each image meridian considered are grouped together and their intensity values averaged. A variance is then computed for each image meridian. This is done for each image meridian tested. Next, the variance and standard deviation of the entire image is computed. If the underlying surface for the image was an LSHGC, then the computed standard deviation of the intensity differences along image meridians must correlate closely with the standard deviation of the normally-distributed noise that was added to the image .[2]

Figure 5 is the image of an LSHGC with a slight rotation around the i axis (towards the viewer). The standard deviation of the normally-distributed noise that was added to the image is 0.02. The computed standard deviation for the image meridians is 0.025, so the image tests positive for an LSHGC.

Figure 6 is a negative example of an LSHGC; the sweep function of the object is definitely not linear. Running the LSHGC test on this image, its computed standard deviation is 0.35, while the standard deviation of the normally-distributed noise actually added to the image is only 0.05. Since the computed standard deviation is not closely correlated with the known degree of noise added to the image, we reject the image as not having been the projection of an underlying LSHGC surface.

The second test determines which images satisfy the reflectance constraint for SHGC's. For this test, the ratio between three meridian points determined by Equation (2.4) should remain constant for all triples taken along an image meridian from the same respective z values.

Consider again the image in Figure 6. This was a negative example of an LSHGC and was rejected by the LSHGC test. We then tested it with SHGC membership test. The underlying Gaussian noise added to the image has known standard deviation of 0.05. Running the SHGC test, we obtain a computed standard deviation of 0.07. So this image is *not* rejected by the SHGC test. Figure 7 is an image of an SGC, not an SHGC, to which Gaussian noise was added with a standard deviation of 0.05. When tested by the SHGC test, the computed standard deviation is 6.75. This image *is* rejected as not being an SHGC (in semi-canonical position).

In Figure 8 a lamp is displayed whose 4 components are SHGC's in approximately canonical position. Gaussian noise was added with standard deviation of 0.05. All components of the lamp successfully passed the SHGC membership test .[3]

[2] The exact determination of whether an image belongs to the class should be made using an F-test. In the examples given above, the *ad hoc* criterion used was if $\hat{\sigma} > 2 \cdot \sigma$ then reject.

[3] The SHGC components were segmented by hand and given to the SHGC membership algorithm separately.

7 Conclusion and Future Work

In conclusion, the reflectance properties of SHGC's and LSHGC's have been studied for various rotational transformations and under various imaging assumptions. Under certain conditions, the intensity images produced by SHGC's and LSHGC's will satisfy certain reflectance constraint equations. These constraints can then be used as a necessary test of membership for a given shape class, like LSHGC's.

Future work involves extending the use of intensity information from SHGC membership tests to solving for actual parameters of the underlying SHGC surface, such as cross-section and sweeping functions.

8 References

[AB73] Agin, G.J. and Binford, T. O., Computer description of curved objects, in *Proceedings of the 3rd IJCAI*, Stanford, California, Aug 20-23, 629-640, 1973.

[BB82] Ballard, D.H., Brown, C.M., *Computer Vision*, Prentice-Hall, Inc., 1982.

[DoC76] do Carmo, M.P., *Differential geometry of curves and surfaces*, Prentice-Hall, Inc., 1976.

[Mar77] Marr, D., Analysis of occluding contour, *Proc. of Royal Society of London* B-197, pp. 441-475, 1977.

[Nal87] Nalwa, Vic, Line-drawing interpretation: Bilateral symmetry, in *Proceedings of the Image Understanding Workshop*, vol. 2, Feb., 1987, pp. 956-967.

[PCM87] Ponce, J., Chelberg, D., and Mann, W., Invariant properties of the projections of straight homogeneous Generalized Cylinders, in *Proceedings of the First International Conference on Computer Vision*, 1987.

[Pon88] Ponce, J., Straight Homogeneous Generalized Cylinders: Differential Geometry and Uniqueness Results, in *Proceedings of Computer Vision and Pattern Recognition Conference*, 327-334, June 1988.

[RM88] Rao, K., and Medioni, G., Useful geometric properties of the generalized cone, in *Proceedings of the Conference on Computer Vision and Pattern Recognition*, June 5-9, 1988, pp. 276- 281.

[Shaf85] Shafer, S., *Shadows and silhouettes in computer vision*, Kluwer Acad Publishers, Boston, 1985.

[UN88] Ulupinar, F., and Nevatia, R., Using Symmetries for Analysis of Shape from Contour, in *Proceedings of the Second International Computer Vision Conference*, 414-426, Dec. 5-8, 1988.

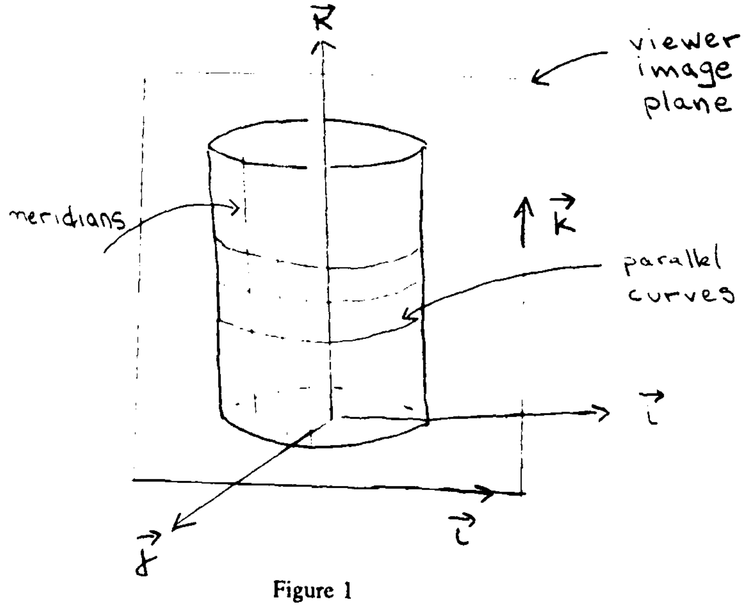
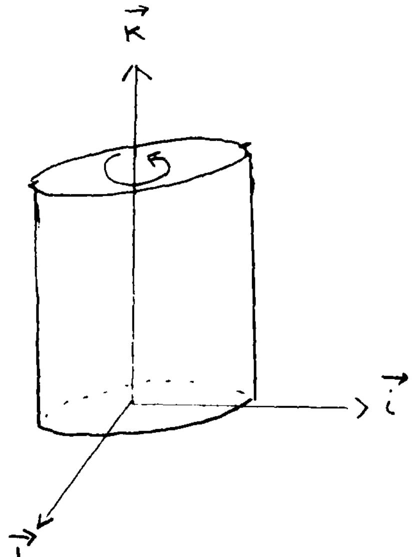


Figure 1



$\vec{j}$ Figure 2

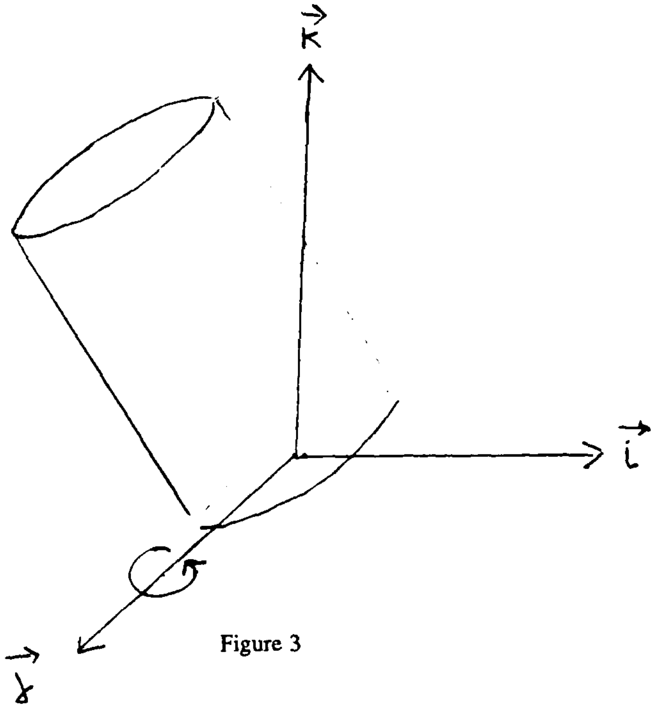


Figure 3

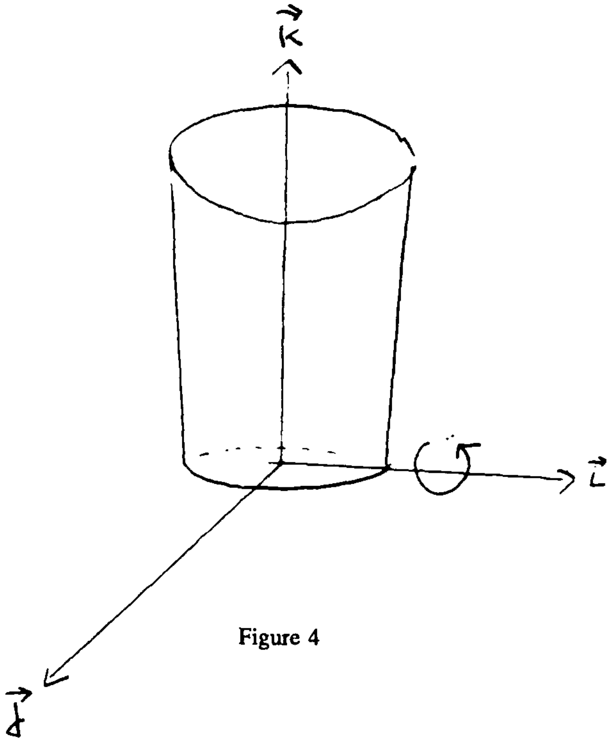


Figure 4

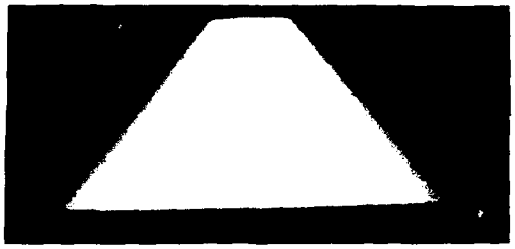


Figure 5

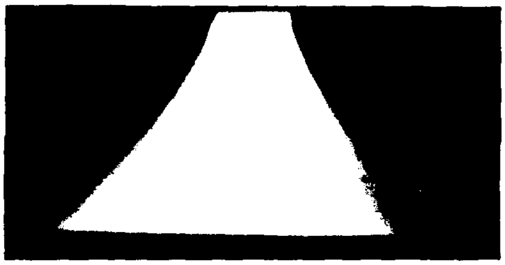


Figure 7

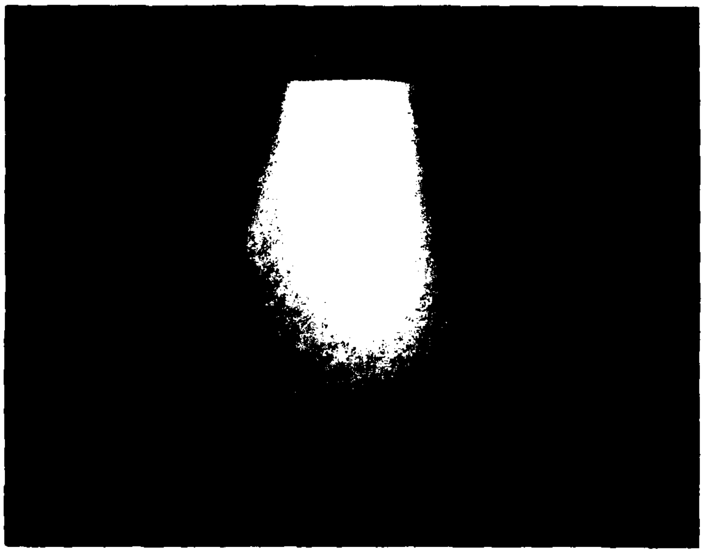


Figure 6

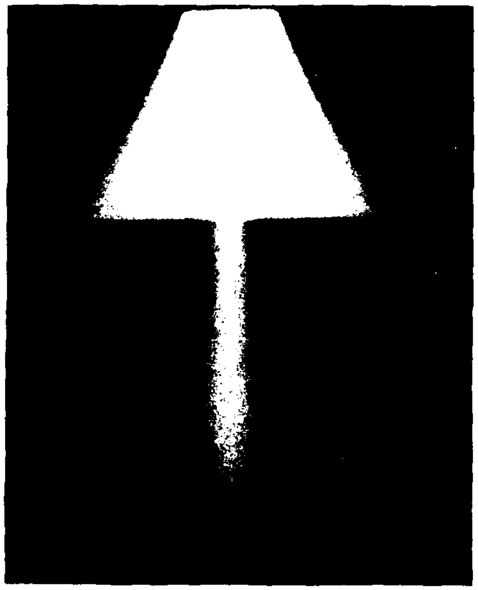


Figure 8

SHAPE UNDERSTANDING FROM LAMBERTIAN PHOTOMETRIC FLOW FIELDS

Lawrence B. Wolff[1]
Computer Science Department
Columbia University
New York, N.Y. 10027

ABSTRACT

A new idea for the analysis of shape from reflectance maps is introduced in this paper. It is shown that local surface orientation and curvature constraints can be obtained at points on a smooth surface by computing the instantaneous rate of change of reflected scene radiance caused by angular variations in illumination geometry. The resulting instantaneous changes in image irradiance values across an optic sensing array of pixels constitute what is termed a *photometric flow field*. Unlike optic flow fields which are instantaneous changes in position across an optic array of pixels caused by relative motion, there is no correspondence problem with respect to obtaining the instantaneous change in image irradiance values between successive image frames. This is because the object and camera remain static relative to one another as the illumination geometry changes.

There are a number of advantages to using photometric flow fields. One advantage is that local surface orientation and curvature at a point on a smooth surface can be uniquely determined by only slightly varying the incident orientation of an illuminator within a small local neighborhood about a specific incident orientation. Robot manipulators and rotation/positioning jigs can be accurately varied within small ranges of motion. Conventional implementation of photometric stereo requires the use of three vastly different incident orientations of an illuminator requiring either much calibration and/or gross and inaccurate robot arm motions. Another advantage of using photometric flow fields is the duality that exists between determining unknown local surface orientation from a known incident illuminator orientation and determining an unknown incident illuminator orientation from a known local surface orientation. The equations for photometric flow fields allow the quantitative determination of the incident orientation of an illuminator from an object having a known calibrated surface orientation.

The main purpose of this paper is to present the new concept of photometric flow fields. Computer simulations will be shown depicting photometric flow fields on a Lambertian sphere, utilizing a point light source. Simulations will be shown depicting how photometric flow fields quantitatively determine local surface orientation from a known incident orientation of an illuminator as well as determining incident illuminator orientation from a known local surface orientation. In planned future experimentation, we intend to implement a more practical configuration using a rotating extended linear light source and computing orientations from relative changes in reflected radiance rather than absolute changes.

1 INTRODUCTION

Vision techniques that rely upon feature information from discontinuities in an image (e.g. edges,corners and lines) quickly breakdown in regions containing smooth surfaces where gray level intensities are generally smoothly varying. It is in these regions of an image that modeling the reflectance properties of an object is most useful. The notion of the *reflectance map* presented in [Horn 1977] is a convenient way of directly relating reflected scene radiance to the surface orientation of a given object material. Such a relation, if physically accurate, can be very useful in obtaining shape information about smooth objects in an image. Two techniques which have paved the way for many other techniques making use of reflectance maps are the method of shape from shading presented in [Horn 1975] and the method of photometric stereo presented in [Woodham 1978]. Both of these methods make the simplifying assumption that the reflectance maps for an object material are Lambertian. A shape from shading method based on local analysis is presented in [Pentland 1984]. Implementation of photometric stereo assuming the presence of specular reflection has been reported in [Ikeuchi 1981] and [Ikeuchi et al. 1986].

A major problem in obtaining local surface orientation at a point on a smooth surface from a single view is the nonuniqueness of surface orientations consistent with the reflected radiance measured from the point. This is because for a given illuminator orientation the reflectance map supplies a single equation constraint between reflected radiance and local surface orientation. By varying the incident orientation of the illuminator, photometric stereo methods obtain additional equation constraints. This is because the reflectance map is dependent on illuminator orientation. However, to get additional equation constraints that are independent in the presence of noise, large variations in incident orientation of the illuminator are required.

This paper introduces another approach to using reflectance maps to obtain relations between local surface orientation and empirically measured photometric quantities. This new approach involves measuring how reflected radiance varies at a point on a smooth surface by varying the angular orientation of the illuminator. By measuring how reflected radiance varies as the illuminator moves in both linearly independent angular directions, two independent equations which constrain local surface orientation are obtained. These newly obtained relations enable the unique determination of local surface orientation from a single view, and angular displacement of a single illuminator within a small neighborhood about a specific incident orientation.

The difference of the reflected radiance at a point on a smooth surface between two slightly displaced incident orientations of an illuminator, divided by the angular displacement, is an approximation to the directional derivative of the reflectance function in the direction of the displacement of the illuminator. This photometric quantity measured at all

[1] This work was supported in part by ARPA grant #N00039-84-C-0165 and NSF grant IRI-88-00370. This work was supported in part by an IBM Graduate Fellowship Award.

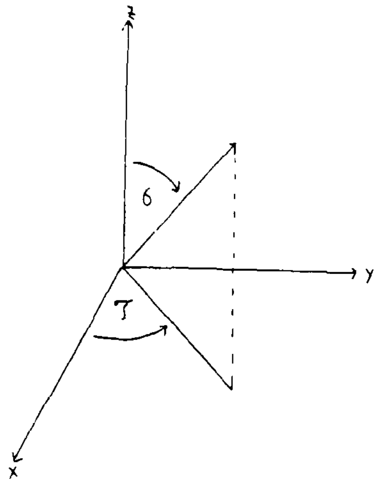


Figure 1:

pixels in an image corresponding to illuminated object points constitutes a *photometric flow field* with respect to the angular displacement of the illuminator in a given direction. The actual directional derivative of a reflectance function in this direction is also a function of local surface orientation. In the case of a Lambertian reflectance function in gradient space representation, the directional derivatives in both angular components of illuminator orientation have the same functional form with respect to gradient space variables p and q representing local surface orientation. These directional derivatives can take on both positive and negative values at points on a surface which are illuminated. Positive values denote an increase in reflected radiance at a point as the light source moves in a given angular direction, and negative values a corresponding decrease in reflected radiance. Except for the fact that the directional derivatives can take on both negative and positive values, they are also functions which are continuously differentiable in variables representing surface orientation and can serve as functions equivalent to reflectance functions with respect to any shape from shading and photometric stereo algorithm. The advantage of evaluating two independent photometric flow fields at a point on a smooth surface should be clear. Other than the surface orientation constraint equation provided by the reflectance function itself, two additional surface orientation constraints are provided from the photometric flow fields.

There is also a big additional advantage to using photometric flow fields which is not possible from conventional use of photometric stereo. This is the duality that exists between the determination of local surface orientation using the known incident orientation of a single illuminator, and the determination of the incident illuminator orientation from a known local surface orientation. Since the evaluation of both independent photometric flow fields at a point on a surface correspond to the same incident illuminator orientation, a known local orientation at that point can be used to determine the incident orientation of the illuminator. This will be demonstrated below.

The reflectance map assumed in this paper is Lambertian. Not only is this reflectance map simple to work with, but in light of recent work which has been successful in isolating the diffuse component of reflection from a variety of material surfaces, Lambertian reflectance is realistic. Reported in [Shafer 1985] and [Klinker et al. 1987] are techniques that separate diffuse and specular components of reflection based upon color analysis. Reported in [Wolff 1989] is an approach to separating diffuse and specular components of reflection using a polarizing filter which isolates polarization components. Since the diffuse component of reflection is Lambertian in nature, even on rough surfaces, Lambertian photometric flow fields can be analyzed on the diffuse component image.

2 LAMBERTIAN PHOTOMETRIC FLOW FIELDS

Using gradient space representation, the reflectance map for a Lambertian surface is given by the function

$$R(p,q) = \frac{p cos\tau sin\sigma + q sin\tau sin\sigma + cos\sigma}{\sqrt{p^2+q^2+1}} \qquad (1)$$

where (p,q) are the gradient space coordinates for the surface orientation, and σ and τ represent the slant and tilt angles for the illuminator (see figure 2). For a given measured value of the reflected radiance, equation 1 becomes the equation for a conic section curve in gradient space. This is referred to in photometric stereo techniques as an *isoreflectance curve* because it gives all possible gradient space representations of surface orientation that are consistent with the measured reflected radiance value.

Suppose now that the incident orientation of the illuminator is shifted in the slant angle σ by a small amount $\Delta\sigma$. The instantaneous rate of change of the reflected radiance in the direction of $\Delta\sigma$ can be approximated by the quantity

$$\frac{\frac{p cos\tau sin(\sigma+\Delta\sigma)+q sin\tau sin(\sigma+\Delta\sigma)+cos(\sigma+\Delta\sigma)}{\sqrt{p^2+q^2+1}} - \frac{p cos\tau sin\sigma+q sin\tau sin\sigma+cos\sigma}{\sqrt{p^2+q^2+1}}}{\Delta\sigma}. \qquad (2)$$

Similarly, the instantaneous rate of change of the reflected radiance in the tilt direction of $\Delta\tau$ is approximated by the quantity

$$\frac{\frac{p cos(\tau+\Delta\tau) sin\sigma+q sin(\tau+\Delta\tau) sin\sigma+cos\sigma}{\sqrt{p^2+q^2+1}} - \frac{p cos\tau sin\sigma+q sin\tau sin\sigma+cos\sigma}{\sqrt{p^2+q^2+1}}}{\Delta\tau}. \qquad (3)$$

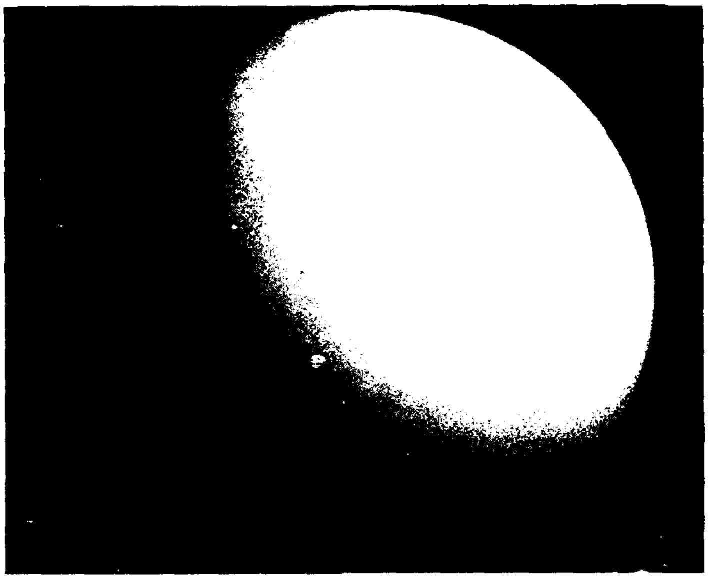


Figure 2:

Consider the expression in equation 1 as a function of the variables σ and τ where p and q are fixed. That is, the reflected radiance at a specific point on an object surface is being viewed as a function of the incident orientation of the light source. Mathematically speaking the expressions in 2 and 3 represent an approximation to the directional derivatives of the function in equation 1 in the unit vector directions ∂_σ and ∂_τ respectively. The vectors ∂_σ and ∂_τ are unit vectors which span the tangent space at a point on the two dimensional manifold of the coordinates (σ, τ). These directional derivatives are computed as follows:

$$\partial_\sigma R(\sigma,\tau) = \frac{\partial R}{\partial \sigma}, \qquad \partial_\tau R(\sigma,\tau) = \frac{\partial R}{\partial \tau}$$

and hence

$$\partial_\sigma R(\sigma,\tau) = \frac{p\cos\tau\cos\sigma + q\sin\tau\cos\sigma - \sin\sigma}{\sqrt{p^2+q^2+1}} \qquad (4)$$

$$\partial_\tau R(\sigma,\tau) = \frac{-p\sin\tau\sin\sigma + q\cos\tau\sin\sigma}{\sqrt{p^2+q^2+1}} \qquad (5)$$

It is possible to take different linear combinations of directional derivatives as follows

$$(a\partial_\sigma + b\partial_\tau)R(\sigma,\tau) = a\frac{\partial R}{\partial \sigma} + b\frac{\partial R}{\partial \tau}$$

where a and b are any real values. This property of directional derivatives can be put to good use in experimental implementation to reduce errors. Moving the illuminator in certain specified directions can create nearly perpendicular intersections of curves in gradient space.

Note that the function in equations 4 and 5 has the same form as in equation 1. Any of these functions set equal to a constant value measured from experiment would generate a conic section in gradient space.

The equations 4 and 5 define Lambertian *photometric flow fields* in positive σ and τ respectively. These can be measured empirically from the expressions in 2 and 3 where the denominator should be the angular variation measured in radians. In a sense Lambertian photometric flow fields are like different Lambertian reflectance functions except that equations 4 and 5 take on negative values as well as positive values.

Figure 2 shows the rendering of a Lambertian sphere using a point light source at incident orientation σ=45 degrees and τ=45 degrees. Figures 3 show a computer simulation of photometric flow fields on the Lambertian sphere, in figure 2, from two independent angular variations of the light source. Figure 3a shows the instantaneous rate of change of image irradiance as the light source orientation changes in positive σ. Figure 3b shows the instantaneous rate of change of image irradiance as the light source orientation changes in positive τ. For figure 3a intensity values northeast of the dark band represent positive reflected radiance changes, while intensity values southwest of the dark band represent negative changes. For figure 3b the positive reflected radiance changes are northwest of the dark band, and negative reflected radiance changes are to the southeast. Figures 4a and 4b show the corresponding relative percentage changes in the reflected radiance produced by the changes represented in figures 3a and 3b respectively. It should be noted that the image intensity values in figures 3 and 4 are relative values. The maximum change over all points on the sphere is set to value 255 with all other values relative to this maximum intensity value. Clearly, maximum values for the photometric flow fields occur at local surface orientations generally far away from the incident orientation of the point light source.

Of immediate notice are the dark bands that are present on the sphere in figures 3 and 4. The intensity values on these bands are very close to zero, and are actually zero along the medial axis of the bands. The medial axes of the dark bands in figures 3 and 4 are examples of *isoflow curves* on the surface of the sphere along which the photometric flow fields are some constant value. In the next section it will be shown that isoflow curves determine local surface orientation constraints. A very simple orientation constraint is that generated by the zero valued isoflow curve which is the medial axis of the dark band in figure 3a. For any points along this isoflow curve the tilt τ of the local surface orientation is equal to the tilt value of the incident orientation of the light source. Most orientation constraints are much more complicated than this.

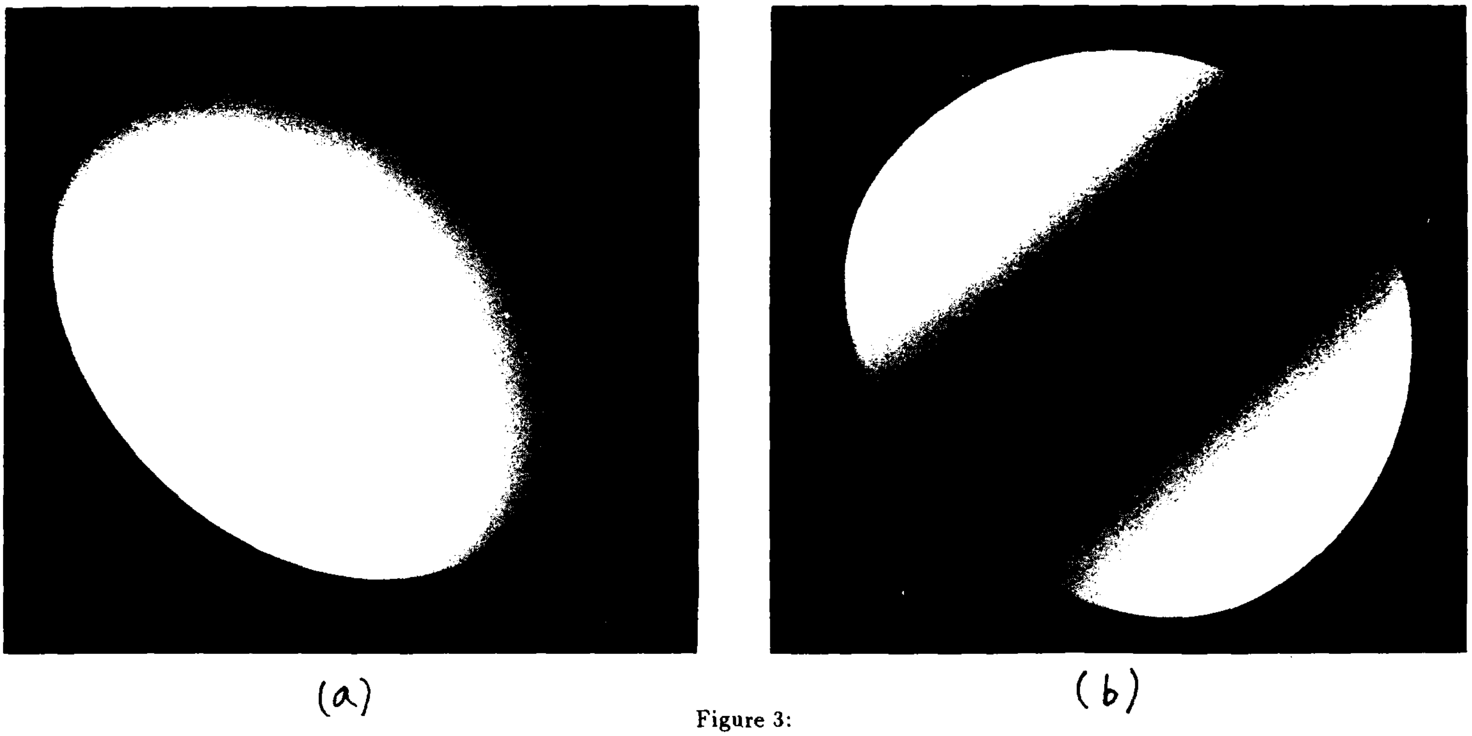


Figure 3:

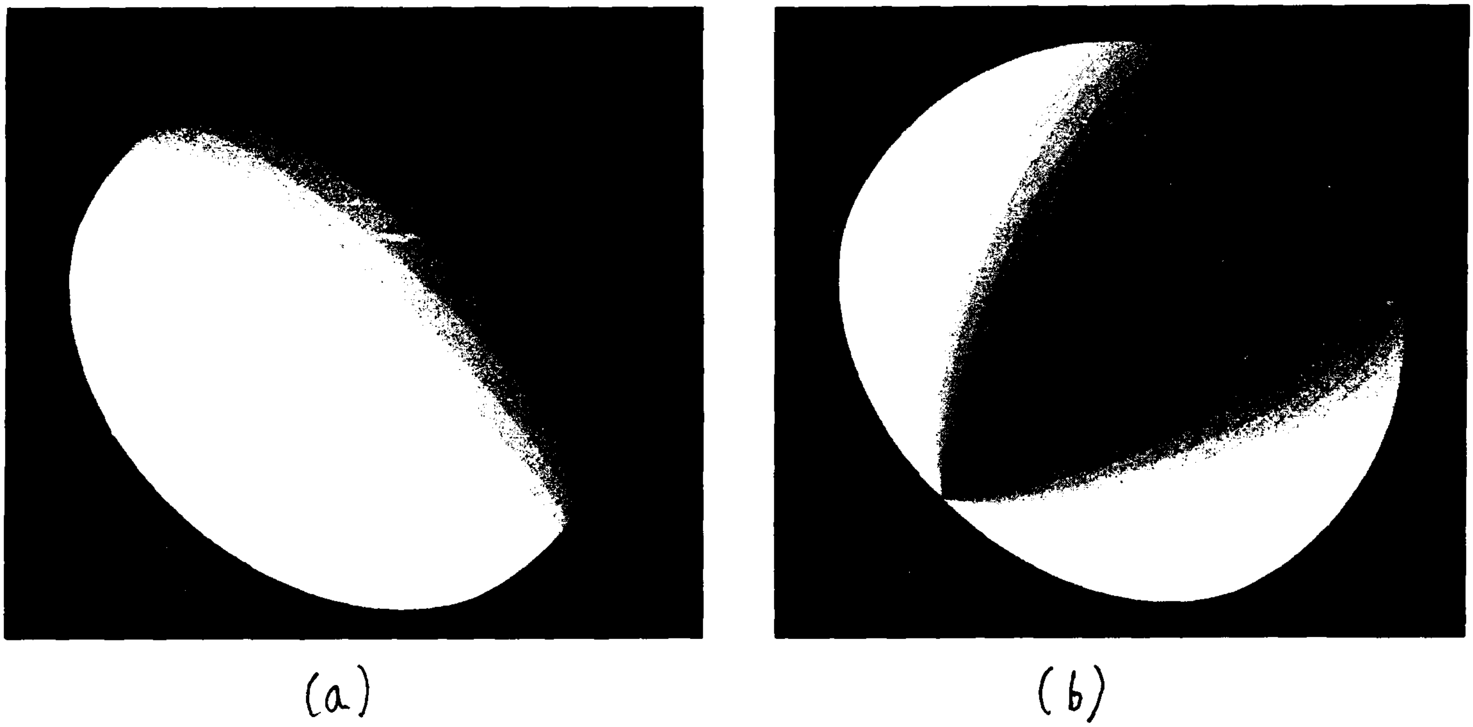


Figure 4:

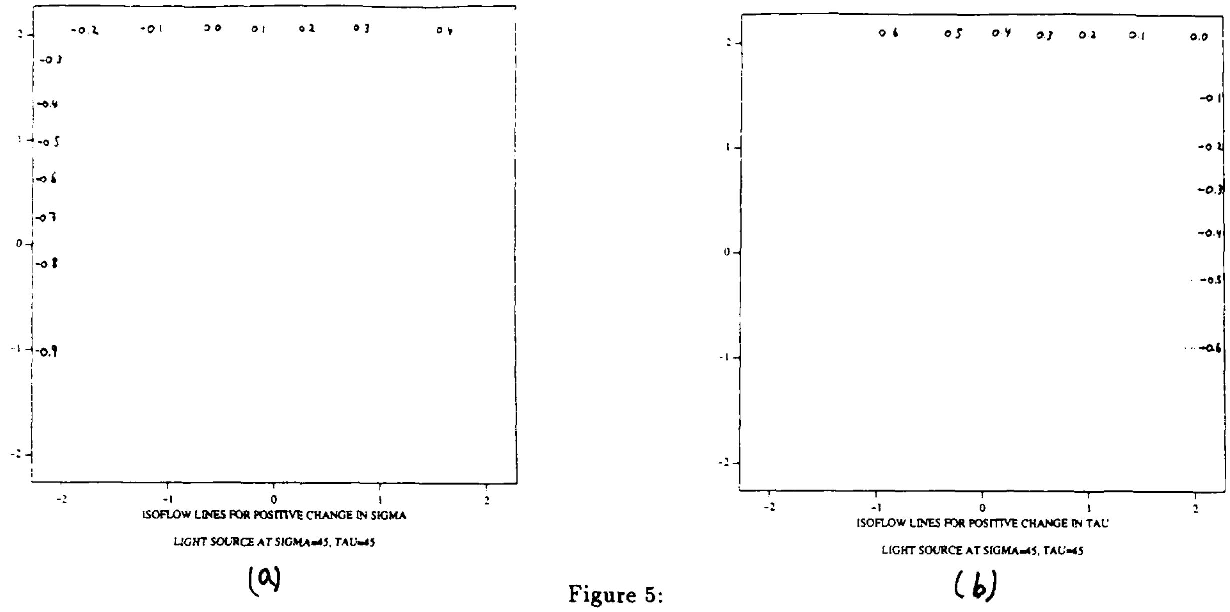


Figure 5:

In a practical experiment where photometric flow fields would be determined from empirical measurement of reflected radiance from a CCD camera, there are two major sources of errors to consider. The first is the measured reflected radiance itself. Enough angular variation in σ and τ needs to be made so that the change in reflected radiance should significantly exceed the repeatability of the photoresponse of pixel sensors. This amount is dictated by the signal-to-noise ratio of the camera being used. Also the output of light sources is not constant over time. The measurements of reflected radiance should be made as quickly as possible between angular variations. The second major source of errors is in the approximation of equations 4 and 5 respectively by expressions 2 and 3.

Again, this paper introduces the basic principles of photometric flow fields and is not complete with full experimental analysis. We simulate below derivations of surface and illuminator orientations based on absolute reflected radiance values. This utilizes the familiar approach of intersecting curves in gradient space. In experimentation it is far more convenient to utilize relative reflected radiance values, and we intend in the future to present experimental results which will build off the basic principles proposed here.

3 SURFACE ORIENTATION FROM PHOTOMETRIC FLOW FIELDS

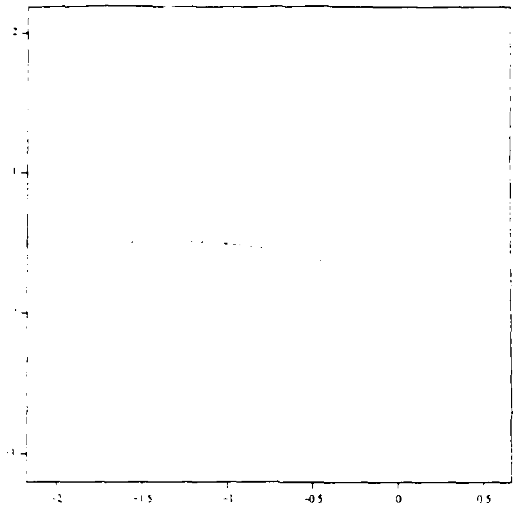
This section will simulate by computer the quantitative derivation of local surface orientation, with the aid of constraints provided from a pair of photometric flow fields. As was seen in the last section, constraints on local surface orientation can be obtained from isoflow curves which are derived by setting equations 4 and 5 to constant values. Examples of isoflow curves in gradient space are shown in figures 5a and 5b with respect to photometric flow fields derived from positive variations in σ and τ respectively. The incident orientation of the light source used is the same as for the simulations in figures 3 and 4 with $\sigma = \tau = 45$ degrees.

The isoflow curves in gradient space depicted in figures 5a and 5b are conic sections of varying eccentricity. The only difference between these isoflow curves and a Lambertian reflectance map depicting isoreflectance curves is that the constant value of photometric flow that an isoflow curve represents can be negative. At a point on a smooth object surface, local surface orientation is constrained by empirically determined values of photometric flow in positive sigma and positive tau which determine the intersection of two isoflow curves in gradient space. Sometimes this results in a unique orientation point, but the rest of the time local surface orientation is only constrained to be at two distinct points. It is not possible to breakup this two point ambiguity by using another photometric flow field obtained by moving the light source in another direction. The reason is that for small angular motions of the light source this direction is simply a linear combination of the motions in σ and in τ. The isoflow line thus obtained would therefore pass through the same two points constraining orientation.

A third constraint curve on local surface orientation in gradient space can be obtained from the isoreflectance curve corresponding to the measured reflected radiance at the point in question. This is simulated in figure 6 for a point on a smooth surface with local surface orientation (-1.0,0.5) in gradient space coordinates. The isoflow curves in figure 6 can be identified by observing the curves in figures 5a and 5b in the same region of gradient space.

Figure 7 depicts the intersection of three isoreflectance curves making small angular variations in the illuminator of 5 degrees in σ and τ. This would be the result of using conventional photometric stereo with very small displacements in the incident orientation of the light source. Under ideal circumstances free of measurement errors this would work fine. Figures 8a and 8b show, in the presence of measurement errors, the comparison of measuring local surface orientation using photometric flow fields with doing this measurement using conventional photometric stereo. These are simulations of worst case errors in the presence of ±5% error in measured reflected radiance. The simulated empirical values for the photometric flow fields are derived by taking the ratios in equations 1 and 2 for 5 degree variations in σ and τ respectively. Even in the presence of approximation error the centroid of the three two-way intersection points in figure 8a produce a measurement error of about 5 degrees, while the measurement error in figure 8b is well over 20 degrees.

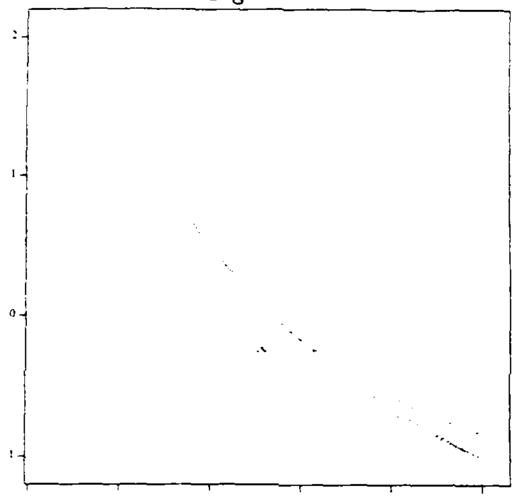
The comparison of the figure pairs 6-8a and 7-8b is to illustrate how orientation errors increase vastly when the intersection of curves in gradient space vary from nearly perpendicular intersections to nearly parallel intersections. Because locally determined isoreflectance curves intersect almost parallel, local photometric stereo techniques are extremely sensitive to camera noise. On the other hand, the intersection of locally determined isoflow curves is not nearly as sensitive, even with inherent approximation error of directional derivatives.



COMPUTING LOCAL SURFACE ORIENTATION FROM
TWO ISOFLOW LINES AND ONE ISOREFLECTANCE LINE

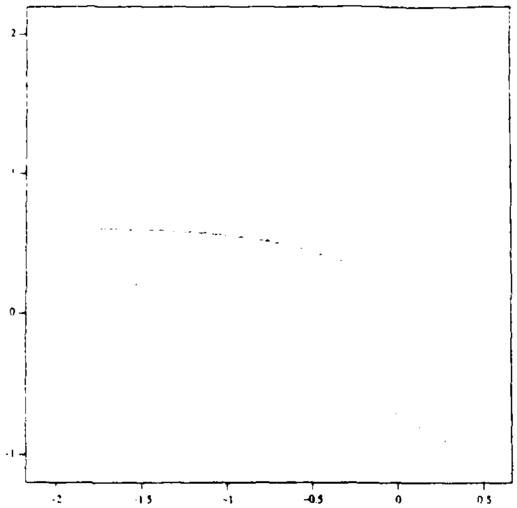
LIGHT SOURCE AT SIGMA=45, TAU=45

Figure 6:



COMPUTING LOCAL SURFACE ORIENTATION FROM
THREE ISOREFLECTANCE LINES
LIGHT SOURCE AT:
SIGMA=45, TAU=45
SIGMA=50, TAU=45
SIGMA=45, TAU=50

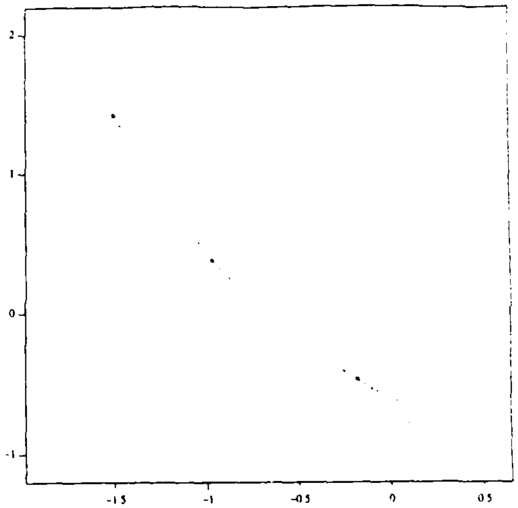
Figure 7:



COMPUTING LOCAL SURFACE ORIENTATION FROM
TWO ISOFLOW LINES AND ONE ISOREFLECTANCE LINE

5% ERROR IN RADIANCE MEASUREMENT

LIGHT SOURCE AT SIGMA=45, TAU=45



COMPUTING LOCAL SURFACE ORIENTATION FROM
THREE ISOREFLECTANCE LINES
LIGHT SOURCE AT
SIGMA=45, TAU=45
SIGMA=50, TAU=45
SIGMA=45, TAU=50

5% ERROR IN RADIANCE MEASUREMENT

Figure 8:

(a)

(b)

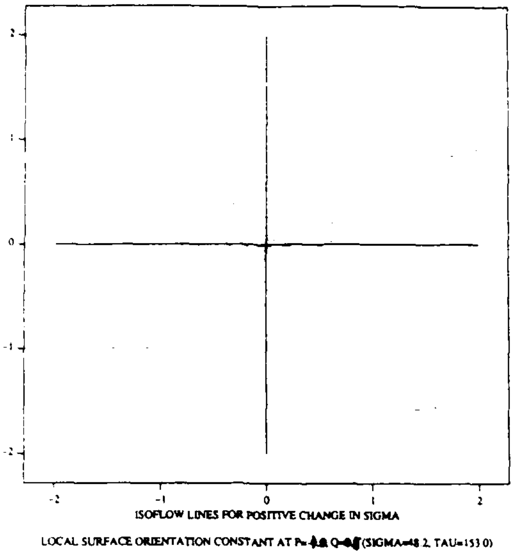


Figure 9:

4 ILLUMINATOR ORIENTATION FROM PHOTOMETRIC FLOW FIELDS

Because photometric flow fields are determined locally there exists a duality between the determination of local surface orientation using the known incident orientation of a light source and the determination of the incident orientation of a light source using the known local surface orientation at a point on a Lambertian surface. This is illustrated by observing equations 4 and 5 and the roles that (p,q) and (σ,τ) play as variables and known constant values. In the last section, local surface orientation was determined from isoflow curves assuming that (σ,τ) were known and constant and (p,q) were variables to be solved. But another interpretation of equations 4 and 5 can be that (p,q) are known, meaning that something like a Lambertian calibration block is used, and the illuminator orientation (σ,τ) are variables to be solved. This is an advantage of using photometric fl.w fields over using conventional photometric stereo since the empirically determined values for the photometric flow fields along with the reflected radiance value all correspond to exactly the same incident orientation of the illuminator.

To demonstrate this duality, a simulation will be performed which reverses the knowns and unknowns of the simulation performed in the last section. Starting with the known orientation value (p,q)=(-1.0,0.5), the simulation in this section will use isoflow curves to determine the unknown incident orientation of the light source (which in actuality is (σ,τ)=(45,45)). To generate curves in gradient space, the variables once again should be in terms of (p,q).. To do this the variables (p,q) in equations 4 and 5 will be held fixed at known values (p_0,q_0) and *the angular representation* (σ,τ) *will* be converted to gradient space representation according to

$$cos\sigma = \frac{1}{\sqrt{p^2+q^2+1}}, \qquad sin\sigma = \frac{\sqrt{p^2+q^2}}{\sqrt{p^2+q^2+1}}$$

$$cos\tau = \frac{p}{\sqrt{p^2+q^2}}, \qquad sin\tau = \frac{q}{\sqrt{p^2+q^2}}$$

whereupon equations 4 and 5 are equivalently

$$\partial_\sigma R(\sigma,\tau) = \frac{p_0p + q_0q - (p^2+q^2)}{\sqrt{p_0^2+q_0^2+1}\sqrt{p^2+q^2}\sqrt{p^2+q^2+1}} \tag{6}$$

$$\partial_\tau R(\sigma,\tau) = \frac{-p_0q + q_0p}{\sqrt{p_0^2+q_0^2+1}\sqrt{p^2+q^2+1}} \tag{7}$$

respectively.

Isoflow curves from variations in τ are once again conic sections in gradient space as can be seen by the form of equation 7. However *isoflow curves from variations in* σ are more complicated. A graphical depiction of isoflow curves produced from setting equation 6 to different constant values is given in figure 9. In figure 10 is the determination of the incident orientation of the illuminator from two isoflow curves from variations in σ and τ respectively, and the isoreflectance curve corresponding to the reflected radiance value at the point on the smooth surface.

5 CURVATURE FROM PHOTOMETRIC FLOW FIELDS

In [Woodham 1978] and [Woodham 1979] a method is presented which determines viewer-centered curvature constraints from shading information. Starting with the image irradiance equation

$$I(x,y) = R(p,q)$$

an application of the chain rule for derivatives yields the matrix equation

$$\begin{pmatrix} \partial I/\partial x \\ \partial I/\partial y \end{pmatrix} = \begin{pmatrix} \partial p/\partial x & \partial p/\partial y \\ \partial q/\partial x & \partial q/\partial y \end{pmatrix} \begin{pmatrix} \partial R/\partial p \\ \partial R/\partial q \end{pmatrix} . \tag{8}$$

The 2x2 matrix in equation 8 represents the Jacobian of the transformation from image coordinates to local surface orientation normals for a given surface. These normals are represented in gradient space coordinates. Hence when this

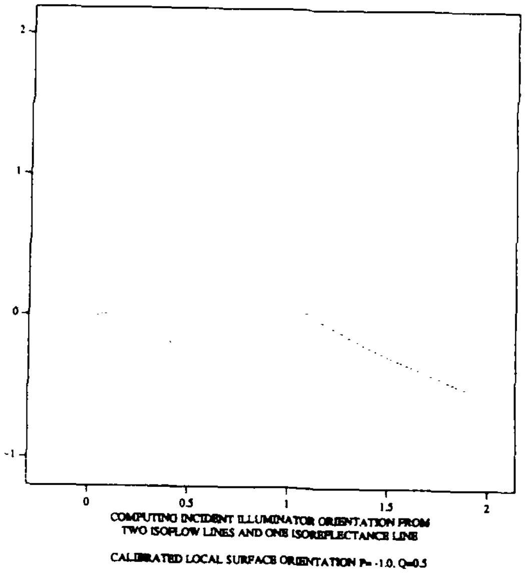


Figure 10:

matrix multiplies an infinitesimal vector change in image coordinates, the resulting vector represents the infinitesimal vector change in the local surface normal of the surface at that point. That is,

$$\begin{pmatrix} dp \\ dq \end{pmatrix} = \begin{pmatrix} \partial p/\partial x & \partial p/\partial y \\ \partial q/\partial x & \partial q/\partial y \end{pmatrix} \begin{pmatrix} dx \\ dy \end{pmatrix} .$$

Hence this Jacobian transformation is termed the *viewer-centered curvature matrix* with respect to image coordinates. On a smooth surface which is parameterized by height above the image plane as $(x, y, f(x, y))$, the viewer-centered curvature matrix is equivalently the Hessian matrix

$$\begin{pmatrix} \partial^2 f/\partial x^2 & \partial^2 f/\partial y^2 \\ \partial^2 f/\partial x^2 & \partial^2 f/\partial y^2 \end{pmatrix} .$$

This utilizes the standard definition of gradient space coordinates as

$$p = \partial f/\partial x \qquad q = \partial f/\partial y .$$

To solve for the three components of the viewer-centered curvature matrix in equation 8 at a point on a smooth surface, the local surface orientation is required to be known in order to compute the gradient of the reflectance map R(p,q). Assuming this is known, equation 8 represents an underconstrained pair of linear equations in three unknowns. In [Woodham 1978] and [Woodham 1979] it is proposed that the underconstrained nature of equation 8 can be ameliorated by solving for certain classes of smooth surfaces. These include developable surfaces for which the determinant of the viewer-centered curvature is always zero, and convex surfaces for which this determinant is greater than zero.

In [Wolff 1987] it is proposed that the viewer-centered curvature matrix can be solved for arbitrary smooth surfaces by combining curvature from shading with photometric stereo. From two additional light source orientations, the two additional matrix equations

$$\begin{pmatrix} \partial I/\partial x \\ \partial I/\partial y \end{pmatrix} = \begin{pmatrix} \partial p/\partial x & \partial p/\partial y \\ \partial q/\partial x & \partial q/\partial y \end{pmatrix} \begin{pmatrix} \partial R'/\partial p \\ \partial R'/\partial q \end{pmatrix} . \tag{9}$$

$$\begin{pmatrix} \partial I/\partial x \\ \partial I/\partial y \end{pmatrix} = \begin{pmatrix} \partial p/\partial x & \partial p/\partial y \\ \partial q/\partial x & \partial q/\partial y \end{pmatrix} \begin{pmatrix} \partial R''/\partial p \\ \partial R''/\partial q \end{pmatrix} . \tag{10}$$

solve for the exact same viewer-centered curvature matrix. This not only obviates the need for assumed auxiliary constraints on the viewer-centered curvature matrix, but in fact overconstrains the equations for better recovery in the presence of measurement error.

The shading information provided by photometric flow fields can equivalently be used to determine the viewer-centered curvature matrix for a smooth surface. That is the reflectance maps $R'(p,q)$ and $R''(p,q)$ in equations 9 and 10 can be replaced by $\partial_\sigma R(p,q)$ and $\partial_\tau R(p,q)$ in equations 4 and 5 respectively. Note however that the image intensity gradients are now photometric flow field gradients in the image plane. In fact two matrix equations resulting from two photometric flow fields obtained from linearly independent angular variations are enough to overconstrain the determination of the three components for the viewer-centered curvature matrix. The solution can be further overconstrained by the use of the original reflectance map using the incident orientation of the light source.

6 CONCLUSION

It has been demonstrated that shape characteristics for smooth surfaces such as local surface orientation and curvature can be derived by examining the instantaneous rate of change in the reflected radiance at points on a smooth surface with respect to angular change in illumination geometry. This rate of change in reflected radiance is determined by the directional derivative of the reflectance function in the angular direction in which the incident orientation of the light source varies. Because the incident orientation of a light source has two angular degrees of freedom, two independent directional derivatives exist giving rise to two independent equations at a point involving local surface orientation.

The approximation to a directional derivative of the reflectance function, in a given direction, is derived at each pixel from two slightly displaced incident orientations of the illuminating light source. This photometric approximation at each pixel in an image corresponding to an illuminated object point constitutes what is termed a *photometric flow field*. At each pixel, the measured value of a photometric flow field constrains local surface orientation at the corresponding object point according to the locus of an isoflow curve in gradient space. This is in the same flavor as constraining local surface orientation from photometric stereo using isoreflectance curves. It was shown that curvature at points on a smooth surface can be obtained from the gradients of photometric flow fields in the image plane.

Because isoflow curves constraining local surface orientation at an object point correspond to the same incident illuminator orientation it was shown that there is a duality between using photometric flow fields to obtain local surface orientation from known incident illuminator orientation and obtaining incident illuminator orientation from known local surface orientation. This duality does not exist for photometric stereo.

The analysis of photometric flow fields was given using variations in angular incidence of a point light source. In practice it is believed that it may be easier to implement photometric flow fields from variations in illumination geometry using an extended source. Consider, for instance, rotating a linear light source (e.g., a fluorescent tube) about its center in a given plane. A photometric flow field would result with respect to angular variation in the plane. Angular variation within two different planes results in two independent photometric flow fields. We hope in the future to implement such a scheme extending the basic results in this paper.

ACKNOWLEDGMENT

The author is grateful to David Kurlander for rendering the images of the Lambertian sphere and the resulting photometric flow fields.

REFERENCES

[Horn 1975] Horn, B.K.P., *Obtaining Shape from Shading Information*, in The Psychology of Computer Vision, P.H. Winston (ed.), McGraw-Hill, pp.115-155, New York, 1975.

[Horn 1977] Horn, B.K.P., *Understanding Image Intensities*, Artificial Intelligence, 8, 1977, pp.1-31.

[Ikeuchi 1981] Ikeuchi, K., *Determining Surface Orientations of Specular Surfaces by using the Photometric Stereo Method*, PAMI, Vol. 3, No. 6, pp.661-669, November 1981.

[Ikeuchi et al. 1986] Ikeuchi, K., Nishihara, H.K., Horn, B.K., et al..., *Determining Grasp Configurations using Photometric Stereo and the PRISM Binocular Stereo System*, The International Journal of Robotics, Vol. 5, No. 1, pp.46-65, Spring 1986.

[Klinker, Shafer and Kanade 1987] Klinker, G.J., Shafer, S.A. and Kanade, T., *Using A Color Reflection Model to Separate Highlights From Object Color*, Proceedings of the First International Conference on Computer Vision, pp. 145-150, 1987.

[Pentland 1984] Pentland, A.P., *Local Shading Analysis*, IEEE trans. on Pattern Analysis and Machine Intelligence, Vol. 6, No. 2, pp. 170-187, March 1984.

[Shafer 1985] Shafer, S.A., *Using Color To Separate Reflection Components*, COLOR research and application, 10(4), pp.210-218, Winter 1985.

[Wolff 1987] Wolff, L.B., *Curvature and Contour from Photometric Stereo*, DARPA image understanding workshop, 1987, pp. 821-824.

[Wolff 1989] Wolff, L.B., *Using Polarization To Separate Reflection Components*, CVPR 1989, These Proceedings.

[Woodham 1978] Woodham, R.J., *Reflectance Map Techniques for Analyzing Surface Defects in Metal Castings*, MIT AI Lab Tech Report AI-TR-457, June 1978.

[Woodham 1979] Woodham, R.J., *Relating Properties of Surface Curvature to Image Intensity*, IJCAI 1979, pp. 971-977.

Image-Flow Estimation: An Information Fusion Approach

Ajit Singh

Department of Computer Science, Columbia University, New York
and
Philips Laboratories, North American Philips Corporation
Briarcliff Manor, New York.

Abstract

Visual motion is commonly extracted from an image sequence in the form of an image-flow field or an image-displacement field. In the past research on measurement of motion fields, three basic approaches have been suggested: matching based approach, gradient based approach and spatiotemporal energy based approach. It has been observed that any single method, based on any one of these approaches works well in a restricted environment. We suggest *information-fusion* as an approach to develop a technique to measure image-flow that could work robustly in a less restricted environment. In this approach, multiple sources give their *opinion* about the measurement of image-flow along with confidence measures. These measurements are then fused on the basis of confidence measures. We propose two levels of fusion. At the *lower* level, various sources of flow-measurement based on a single approach (of the above three) are called upon to provide a constraint on the flow, along with a confidence measure and the constraints are then combined according to their confidences. At the *higher* level, measurements from different approaches are combined in order to overcome their individual shortcomings. In this paper, we report our work on lower-level fusion within *matching based* methods and outline our plan for future research of higher-level fusion of matching based methods and gradient based methods.

1 Introduction

Image motion is a major source of three dimensional information about a scene which is in motion relative to the observer [17, 5]. Image motion is commonly represented by a dense image-flow-field (or an image-displacement-field) that assigns a two-dimensional velocity (or displacement) vector to every point on the visual field[1]. Image flow has been shown to contain information about the three dimensional structure and motion of the scene [12, 18]. However, the image flow must be reliably extracted from the time varying imagery (typically a sequence of images taken in quick succession) in order to be used for three dimensional scene interpretation.

The techniques to compute image-flow from a sequence of images follow one of the thr?e basic approaches: (i) matching based approach [3, 6, 13], (ii) gradient based approach [10, 11, 7] and (iii) spatiotemporal energy based approach [1, 8]. A review and a comparative study of these can be found in [3, 14, 2]. All of these techniques are computationally local in nature, i.e., they make use of intensity information in a small neighborhood of a point to compute the image-velocity vector at that point. It is well known that local information is insufficient to compute the complete image-velocity at a point. More specifically, if the local intensity function in the image has low gradient in some direction, then the component of image-motion in this direction cannot be recovered from the local measurements alone. This is referred to as the aperture problem [10, 9]. Typical solutions to the aperture problem invoke either a smoothness constraint [10, 19, 9, 11, 3] or the analytical structure of image flow [18, 16]. Singh [15] does a comparative study of various techniques.

It has been observed that any single technique, based on any one of the above three approaches tends to work well only in a restricted environment. For example, matching based approaches give reliable estimates of image-flow in those spatial regions in the image that are "well structured", i.e., that have rather distinct features such as intensity-corners intensity edges with high curvature etc. On the other hand, gradient based approaches give reliable

[1] In this paper, we will use the terms flow-field and displacement-field interchangeably.

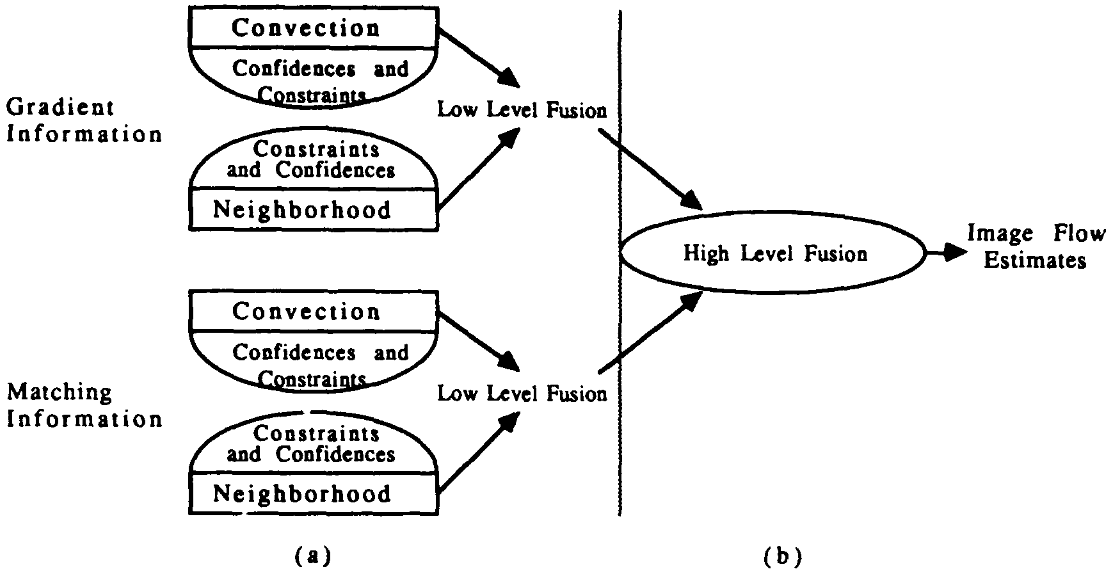


Figure 1: Two Levels of Information Fusion

estimates in those spatial regions that have sufficient intensity gradient but not an intensity discontinuity *and* where the local image displacement is small compared to the local spatial structure. Spatiotemporal energy based methods tend to work well in textured regions. Therefore, intuitively, one can say that a technique that *intelligently* combines the measurements obtained by using more than one of the above approaches can give a reliable estimate over a broad range of environments. What's required to develop such a technique is a framework that can formulate any approach to give not only the flow measurement at each point in the image, but also a confidence measure on how reliable the measurement is. The confidence measure can serve as a basis for intelligent fusion of the estimates obtained from multiple approaches. We call this *high-level* fusion. Figure 1b depicts the underlying concept of high-level fusion.

Furthermore, within any approach, there can be multiple sources of information about image-flow. For example, in a technique based on gradient based approach, such as that of Horn and Schunck [10], two sources of information are used: the spatial/temporal intensity gradients and the local variation of image velocities. Each of these sources provides one constraint on the image-flow. The constraints are simultaneously solved for each point on the image to give the flow estimates. We suggest that the multiple (two in the current example) sources within an approach should be captured in such a formulation that gives *not just a constraint, but a constraint and a confidence measure*, which could be intelligently combined in a manner described above. The recent algorithm suggested by Scott [13] that uses a matching based approach does in fact work along these lines and has shown a definite promise.

We have come up with the following generalization for *matching based* and *gradient based* approaches[2]. *In each of these approaches, two classes of constraints on image-flow can be derived along with their confidence measures:* (i) *convection constraints* and (ii) *neighborhood constraints*. These constraints can be fused on the basis of confidence measures to give the flow estimates. We call this *low-level* fusion. Figure 1a depicts the underlying principle of low-level fusion.

In section 2 of this paper we briefly review some of the representative formulations reported in the past research to analyze their individual shortcomings. This review is intended to serve as a motivation for the concept of fusion, rather than a criticism. In section 3, we illustrate the concept of low-level fusion by describing a framework that uses a matching based approach and show results of implementation on real imagery. In section 4, we give a brief outline for derivation of an identical framework for a gradient based approach. We also describe our plans for future research on high-level fusion of these two approaches.

2 Some Representative Examples From Past Research

Horn and Schunck [10] used a gradient based formulation of the image flow problem. They used the assumption that the intensity I of a point in a moving pattern remains constant over time to derive one constraint on the image-motion. They used the assumption of smoothness of the flow-field to impose another constraint to solve the

[2] Whether this generalization applies to spatiotemporal energy based approach is currently under investigation.

aperture problem. This formulation leads to the following difficulties. (i) it tends to blur the motion field at the genuine motion boundaries. and (ii) no advantage is taken of the fact that there are points in the visual field, such as corners, where the aperture problem does not exist and the true image-velocity is completely known. The formulation used to derive the motion constraint is incapable of extracting *all* the information available at a corner. Nagel [11] developed a technique that used second order variations of image-intensity to captuie the information available at the points of high curvature such as corners etc. and used a smoothness constraint that is enforced strongly along the direction of the underlying contour and weakly across the contour. This formulation does not require any a priori knowledge about the location of the contours, but has practical limitations. It is based on the second order spatial partial derivatives of the image intensity that are hard to estimate in real imagery.

Hildreth [9] used a smoothness criterion in which the velocity field is required to be smooth only *along* a contour and not across it. This formulation suffers from the following drawbacks. (i) It gives the true image-flow only at the points that lie along contours, and not everywhere on the visual field, (ii) this formulation, like that of Horn and Schunck, does not take any direct advantage of the fact that points of strong curvature along a contour have more information than just the normal flow, (iii) no consideration is given to the fact that contours can also be due to albedo transitions and texture, where the flow information *should* be propagated across the contour to provide an additional constraint on the velocities of intervening areas and (iv) in presence of multiple objects moving with different velocities, the method provides no guarantees of not propagating velocities along two contours that intersect but belong to two different objects.

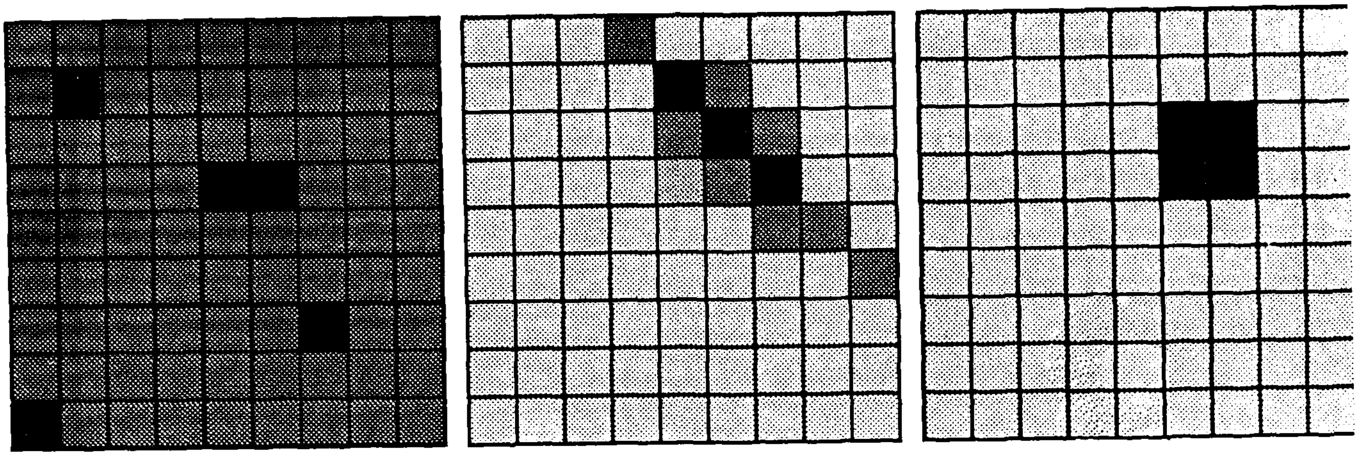
Anandan used a hierarchical matching based approach [3]. In his approach, a search window around the original location of a pixel (or feature) is searched for the "best match" that describes the new location of the pixel (or feature), thus giving the image-displacement. The principle-curvatures of the matching-strength "surface" at the location of best match are taken to be the confidence measures in two orthogonal directions. Anandan used a modified version of Horn and Schunck's smoothness criterion. Recently, there has been significant effort in using some assumptions about the local geometry of the surfaces whose motion causes the image-flow to measure image-flow [12, 18, 16]. These techniques assume that the location of flow-discontinuities is known a priori. This assumption requires that a reasonable high level image segmentation has been done (beyond a simple zero-crossing analysis). This is an unrealistic assumption, particularly from a physiological viewpoint, given that visual motion measurement is an early process. It has been suggested [18] that the model (that formulates the image-flow computation) itself can be used to determine the boundaries of analyticity, but in presence of noise, such a scheme suffers from practical limitations.

From the above review it can be summarized that the any individual method suffers from one or more of the following three drawbacks: (i) it tends to blur the flow-field across motion boundaries, (ii) it requires the second derivatives of the flow-field, which are hard to estimate from noisy image data, (iii) it assumes that image segmentation has been done, thus requiring a lot of global information.

It appears from the above discussion that a good method to compute image-flow (i) must *generate* the flow boundaries rather than require a-priori information about them and (ii) must make use of all the information that is available, for example, at corners and such points of high second order variation and (iii) give a confidence measure that reflects the reliability of flow-estimate at every point. In the following section, we propose a technique that applies low-level fusion described earlier to successfully achieve these three objectives. This technique is a generalization of the recently published 4-line algorithm of Scott [13].

3 Low Level Fusion in a Matching Based Approach

According to the generalization proposed at the end of section 1, there are *two* sources of information about image-flow in a local neighborhood: (i) convection constraints and (ii) neighborhood constraints. *Convection* refer to the information available from the fact that when a feature under motion is observed in images taken at different time instants, some properties of the feature remain *invariant*. For example, in the approach proposed by Horn and Schunck, this property is taken to be the intensity of the pixel (feature) in consideration. *Neighborhood* information refers to the distribution of the local flow vectors in a region in the image. In this section, we describe a technique to (i) derive the above two classes of constraints along with confidence measures using a matching based approach and (ii) fusing the constraints to recover image flow. According to the definition given earlier, this is an instance of low-level fusion.



(a) In a uniform region (b) Near an edge (c) Near a corner

Figure 2: Matching Strength Surface for some Representative Examples

3.1 Deriving the Convection Constraints

We first deal with the issue of deriving convection constraints at a qualitative level. After having established the underlying principles of our approach qualitatively, we will present the necessary mathematical framework to translate our apnroach into an algorithm. A matching based approach such as [3] essentially involves an explicit search for the best match for a given pixel of an image in the subsequent images of the sequence. Typically, such a search is conducted by correlation-based techniques. Most of the techniques suggested in the prior research typically select the pixel (in the second image) with the best matching strength as the *match* for the pixel (in the first image) under consideration. It is obvious that such a selection is not always possible. For instance, in regions of uniform intensity, the matching strength is likely to be quite uniform over the search window. Similarly, in the neighborhood of an intensity edge, there will be several points of high matching strength, distributed along a curve. In general, the distribution of the matching strength over the search window will depend on the distribution of the intensity variation in the underlying image. Thus, the search area can be visualized as covered with a "matching strength surface". The information contained in the matching strength surface can be characterized quantitatively by a principal axis decomposition of the surface. The natching strength at a point serves as the "mass" at the point. Intuitively, we can say that the principal axis decomposition characterizes the information that we discussed above in the first motivating factor. This characterization was used in [13].

The normalized moment of inertia about each of the two principal axes is inversely related to the confidence in assuming that the motion lies on that axis. This can be observed for some representative cases as shown in Figure 2. In areas where the intensity distribution is uniform, (Figure 2a) the matching strength will be high everywhere in the search window $\mathcal{W}_s$. This will result in a high normalized moment of inertia around both the principal axes, and thus a low confidence in assuming that the motion lies along any one of the two axes. Similarly, in the vicinity of an intensity edge, the pixels with a high matching strength will be oriented along a curve in the search window $\mathcal{W}_s$ (see Figure 2b). This will result in a high moment of inertia about the minor principal axis and a low moment of inertia about the major principal axis. Consequently, there will be low confidence in the motion lying on the minor axis and a high confidence in the motion lying on the major axis. This observation is consistent with the aperture problem. Finally, the case of a "corner" in the intensity image is shown in Figure 2c. The strong matches are concentrated around a single point in the search window $\mathcal{W}_s$, giving a low moment of inertia about both the principal axes. As a result, there is a high confidence in the motion lying on both the principal axes, that is, on their intersection.

Thus, we have the barebones of a procedure to capture the convection information in form of two linear constraints for each pixel, along with their associated confidence measures. We refer to these lines as the *convection constraint lines*. As we have seen in the qualitative description above, the two essential steps in computing the convection constraint lines (in case of a matching based approach) are: (i) computing the matching strength for each pixel, over the search area that corresponds to that pixel and (ii) performing a principal axis decomposition of the matching strength surface to compute the two convection constraint lines and their confidences.

We will now present the quantitative framework that essentially performs the two steps mentioned above. We work with $2 * T + 1$ images taken at time instants $t = -T$, $t = -T + 1$, ... through $t = +T$ to estimate image flow field that corresponds to the image taken at time $t = 0$. We refer to the image taken at time t by I_t For purpose of simplicity we assume that $T = 1$ (i.e., the total number of images is three) and that the displacement of a pixel over $2 * T + 1$ time units is limited to N pixels in any direction. This restriction can easily be relaxed by using a hierarchical approach [3]. Finally, we assume that the time elapsed between two successive images is unity. In this case, the displacement-field coincides with the flow field. What we compute explicitly is the displacement field. For this purpose, we will be using a $\delta x - \delta y$ plane rather than the $u - v$ plane to represent the motion constraint lines. In this new scheme, δx and δy refer to the x and y components of the displacement vector for a pixel.

Computing the Matching Strength: Several *match measures* between two intensity patterns have been proposed in the literature. The most commonly used ones are direct correlation, mean normalized correlation, variance normalized correlation, sum of squared differences and sum of absolute differences, etc. Burt, Yen and Xu [6] and Anandan [3] provide a comparative survey of the various measures. In accordance with the reasoning of [3] we choose the sum of squared differences, SSD, as a measure of the match. The smaller the SSD, the better the match. Since we need a matching strength that *increases* with the quality of match, we use the reciprocal of the SSD as the matching strength.

For each pixel $\mathcal{P}(x, y)$ at the location (x, y) in the image I_0 taken at time $t = 0$, a window $\mathcal{W}_p$ of size $(2 * n + 1) \times (2 * n + 1)$ is formed around the pixel. The search window $\mathcal{W}_s$ of size $(2 * N + 1) \times (2 * N + 1)$ is established around the pixel at location (x, y) in all other images. The $N \times N$ matrix depicting the matching strength surface $\mathcal{M}(\delta x, \delta y, t)$ between $\mathcal{W}_p$ and a similar $(2*n+1) \times (2*n+1)$ window around *each* pixel in $\mathcal{W}_s$ in image I_t, displaced from (x, y) by an amount $(\delta x, \delta y)$, is computed as:

$$\mathcal{M}(\delta x, \delta y, t) = \frac{1}{\sum_{i=-n}^{n} \sum_{j=-n}^{n} (\mathcal{I}_0(x + i, y + j) \; - \mathcal{I}_t(x + \delta x + i, y + \delta y + j))^2}$$

$$-N \leq \delta x, \delta y \leq +N \qquad (1)$$

In this expression, $\mathcal{I}_0(x, y)$ and $\mathcal{I}_t(x, y)$ refer to the pixel intensities at the location (x, y) in the image I_0 and I_t respectively. The correct matching strength surface $\mathcal{M}(\delta x, \delta y)$ for the pixel under consideration is found by "reversing" (in both δx and δy directions), the matching strength surface $\mathcal{M}(\delta x, \delta y, -1)$ and adding it to $\mathcal{M}(\delta x, \delta y, +1)$[3]. This measure is different from that used by Scott [13] in that opinion of more than one pair of images is being sought.

Deriving the Convection Constraint Lines and Their Confidences: The two convection constraint lines described above are to be derived for *each* pixel $\mathcal{P}(x, y)$ along with their confidence measures. These constraint lines are the principal axes of the Matching Strength Surface $\mathcal{M}(\delta x, \delta y)$ for the pixel under consideration, as derived above in Equation 1 (with the matching strength at a point serving as the "mass"). The principal axes can be determined as the eigen vectors of the *scatter matrix* that corresponds to the distribution of matching strength over the search area and is characterized as:

$$\mathcal{S} = \begin{pmatrix} \sum_{\delta x} \sum_{\delta y} \mathcal{M}(\delta x, \delta y)(\delta x - \delta x_c)^2 & \sum_{\delta x} \sum_{\delta y} \mathcal{M}(\delta x, \delta y)(\delta x - \delta x_c)(\delta y - \delta y_c) \\ . & . \\ \sum_{\delta x} \sum_{\delta y} \mathcal{M}(\delta x, \delta y)(\delta x - \delta x_c)(\delta y - \delta y_c) & \sum_{\delta x} \sum_{\delta y} \mathcal{M}(\delta x, \delta y)(\delta y - \delta y_c)^2 \end{pmatrix} \qquad (2)$$

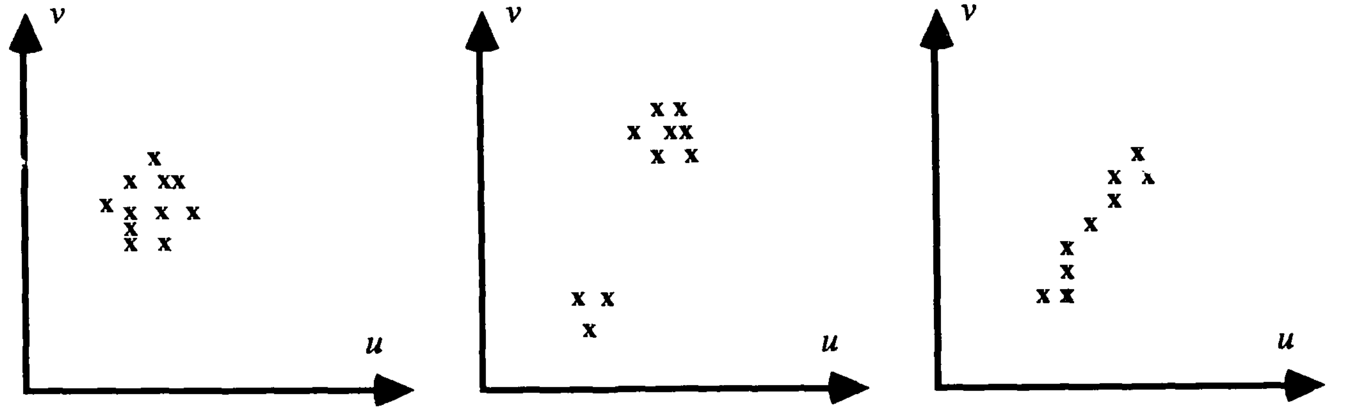
In this definition, the summation is carried out over $-N \leq \delta x, \delta y \leq +N$. Also, $(\delta x_c, \delta y_c)$ denotes the "center of mass" of the cloud of matches. The eigen vectors of the scatter matrix give two convection constraint lines [4] of the form:

$$\mathcal{L}_1 : a_1 \delta x + b_1 \delta y + c_1 = 0$$

$$\mathcal{L}_2 : a_2 \delta x + b_2 \delta y + c_2 = 0 \qquad (3)$$

Their associated confidences $\mathcal{C}_1$ and $\mathcal{C}_2$ can easily be computed as the the reciprocal of the normalized moment of inertia of the matching strength distribution about these two lines. The line $\mathcal{L}_1$, corresponding to the major principal

[3] When using more than three images, the simple addition is replaced by a weighted-sum where the weight reflects the time separation of I_t from I_0.



(a) Uniform Neighborhood (b) Flow Boundary (c) Gradual Change of Depth

Figure 3: Neighborhood Flow Distribution for some Representative Examples

axis captures the same physical meaning as the motion constraint line of Horn and Schunck [10] but does it in a more unrestricted way. For example, the restriction that the neighborhood should not span an intensity discontinuity, which is very crucial for Horn and Schunck's approach, does not have to be met in the formulation given above.

3.1.1 Deriving the Neighborhood Constraints

As in the case of the derivation of convection constraints, we will deal with the neighborhood information on a qualitative level first before we go into a quantitative description of the method. Let us assume for a moment that the correct flow-field for an image sequence is available (from some magical technique). What can we say about the distribution of flow variations in a neighborhood? We can map the flow vectors in a neighborhood to points in the $u - v$ plane. Some representative cases are shown in Figure 3. In a neighborhood where the flow-field is uniform, all the flow-vectors, when mapped to the $u - v$ plane, will be clustered around a single point as shown in Figure 3a. Similarly, if a neighborhood spans a flow boundary (between two regions of more or less uniform flow fields), the flow vectors, when mapped to the $u - v$ plane will cluster around two distinct points, as shown in Figure 3b. Finally, in the case of a gradual change of depth of a moving rigid object, where the flow-field gradually changes from one uniform distribution to another, the flow vectors, when mapped to the $u - v$ plane, could be oriented as shown in Figure 3c. Although we have used the "correct" flow-field to gain some intuition about the distribution of flow variations in a neighborhood, the conclusions will nevertheless be true during an iterative update process for computing the flow-field, provided the iteration is approaching convergence.

Scott [13] reports that principal axis decomposition (of the flow-field in a neighborhood, mapped onto the $u - v$ plane) can serve as a good quantitative description of the distribution. In this case, a point in the $u - v$ plane (corresponding to the flow vector of a given pixel in the neighborhood) is assigned a "mass" that depends on the distance of this pixel from the "center" of the neighborhood. We assign these masses according to a Gaussian mask. The principal axis decomposition gives two motion constraint lines for each pixel, along with an associated confidence. These two lines reflect the *opinion of the neighborhood* about the velocity of a pixel that lies in its center, as opposed to the two motion constraint lines derived by matching that refer to a pixel's *own opinion* about its velocity. We call these lines the *neighborhood constraint lines*.

The quantitative derivation of the neighborhood constraint lines is quite straightforward. Once again, we assume that the time elapsed between two successive images is unity so that the $u - v$ plane coincides with the $\delta x - \delta y$ plane. We assume that a neighborhood $\mathcal{N}$ around a pixel $\mathcal{P}(x, y)$ whose opinion is used in deriving the neighborhood constraints is of size $(2 * k + 1) \times (2 * k + 1)$. The displacement vector at each of these pixels is mapped to a point in the $\delta x - \delta y$ plane and assigned a "mass" that depends on the distance of the pixel from $\mathcal{P}(x, y)$ according to a Gaussian mask (similar to the approach used by Horn and Schunck [10]). The principal axis decomposition can be done using a technique identical to that used for convection constraint lines. The Neighborhood Constraint Lines thus obtained have a form:

$$\begin{aligned} \mathcal{L}_3 &: a_3\delta x + b_3\delta y + c_3 = 0 \\ \mathcal{L}_4 &: a_4\delta x + b_4\delta y + c_4 = 0 \end{aligned} \tag{4}$$

Their associated confidences $\mathcal{C}_3$ and $\mathcal{C}_4$ can be computed as the the reciprocal of the moment of inertia of the "assigned mass" about these two lines.

3.2 Fusing the Constraints

The fusion process would involve finding a velocity at each point that satisfies *all* the constraints available at that point to a degree suggested by their confidence measures. A rigorous approach to achieve this is currently under investigation. In this work, we select a point in the $u - v$ plane that minimizes the sum of its squared distances from each of the constraint lines, weighted by its confidence measure, and accounting for the relative faith in the convection constraint lines versus the neighborhood constraint lines. Thus, the quantity to be minimized is given by:

$$\mathcal{F} = (1-\psi^2)((\mathcal{L}_1\mathcal{C}_1)^2 + (\mathcal{L}_2\mathcal{C}_2)^2) + \psi^2((\mathcal{L}_3\mathcal{C}_3)^2 + (\mathcal{L}_4\mathcal{C}_4)^2) \tag{5}$$

The role of ψ is similar to that of the *smoothing factor* σ used by Horn and Schunck [10] or Anandan [3]. It reflects our a priori relative confidence in the convection constraints and the neighborhood constraints. If ψ is chosen close to zero, the resultant motion field will adhere closely to that suggested by the convection constraints. On the other hand, if ψ is chosen close to unity, the neighborhood opinion will strongly bias the motion field. Our choice of the formulation of Equation 5 is motivated by that of Scott [13]. However, in our implementation, a small value of ψ is chosen in the beginning, and as the iterations proceed, the value of ψ is gradually increased as the faith in the opinion of the neighborhood increases.

The problem of finding the minimum of the function $\mathcal{F}$ in Equation 5 is that of finding an unconstrained minimum of a quadratic function of two variables δx and δy. It can be shown via critical point analysis that such a minimum, if it exists is given by:

$$\begin{aligned} \delta x &= \frac{t*r - q*s}{p*q - r^2} \\ \delta y &= \frac{s*r - t*p}{p*q - r^2} \end{aligned} \tag{6}$$

where:

$$\begin{aligned} p &= (1-\psi^2)(\mathcal{C}_1^2a_1^2 + \mathcal{C}_2^2a_2^2) + \psi^2(\mathcal{C}_3^2a_3^2 + \mathcal{C}_4^2a_4^2) \\ q &= (1-\psi^2)(\mathcal{C}_1^2b_1^2 + \mathcal{C}_2^2b_2^2) + \psi^2(\mathcal{C}_3^2b_3^2 + \mathcal{C}_4^2b_4^2) \\ r &= (1-\psi^2)(\mathcal{C}_1^2a_1b_1 + \mathcal{C}_2^2a_2b_2) + \psi^2(\mathcal{C}_3^2a_3b_3 + \mathcal{C}_4^2a_4b_4) \\ s &= (1-\psi^2)(\mathcal{C}_1^2a_1c_1 + \mathcal{C}_2^2a_2c_2) + \psi^2(\mathcal{C}_3^2a_3c_3 + \mathcal{C}_4^2a_4c_4) \\ t &= (1-\psi^2)(\mathcal{C}_1^2b_1c_1 + \mathcal{C}_2^2b_2c_2) + \psi^2(\mathcal{C}_3^2b_3c_3 + \mathcal{C}_4^2b_4c_4) \end{aligned} \tag{7}$$

In the algorithm that we propose, the convection constraint lines are computed only once at the onset and the neighborhood constraint lines are computed once for each iteration. In a simple implementation, the algorithm was applied to a sequence of three 64 × 64 images of a polyhedral block in motion, with the correlation window and the search window of sizes 3 × 3 and 5 × 5 respectively. The middle frame of the sequence and the flow-field are shown in figure 4. For sake of clarity, a zoomed version of a section of the flow-field is also shown. Over the three frames, the block was rotated (in a plane on which it rested) clockwise about a point that was close to the lower left hand corner of the image. The maximum linear displacement of a point on the block was limited to about three pixels in the image. It is difficult to compare the results to the "real flow" because we do not know the real flow. We have run the algorithm on about half a dozen image-sequences taken in the lab and outdoors, and the results appear quite convincing visually. We have also applied the algorithm to synthetic imagery, where the "real flow" was known and have recovered the image flow correct to within five percent everywhere.

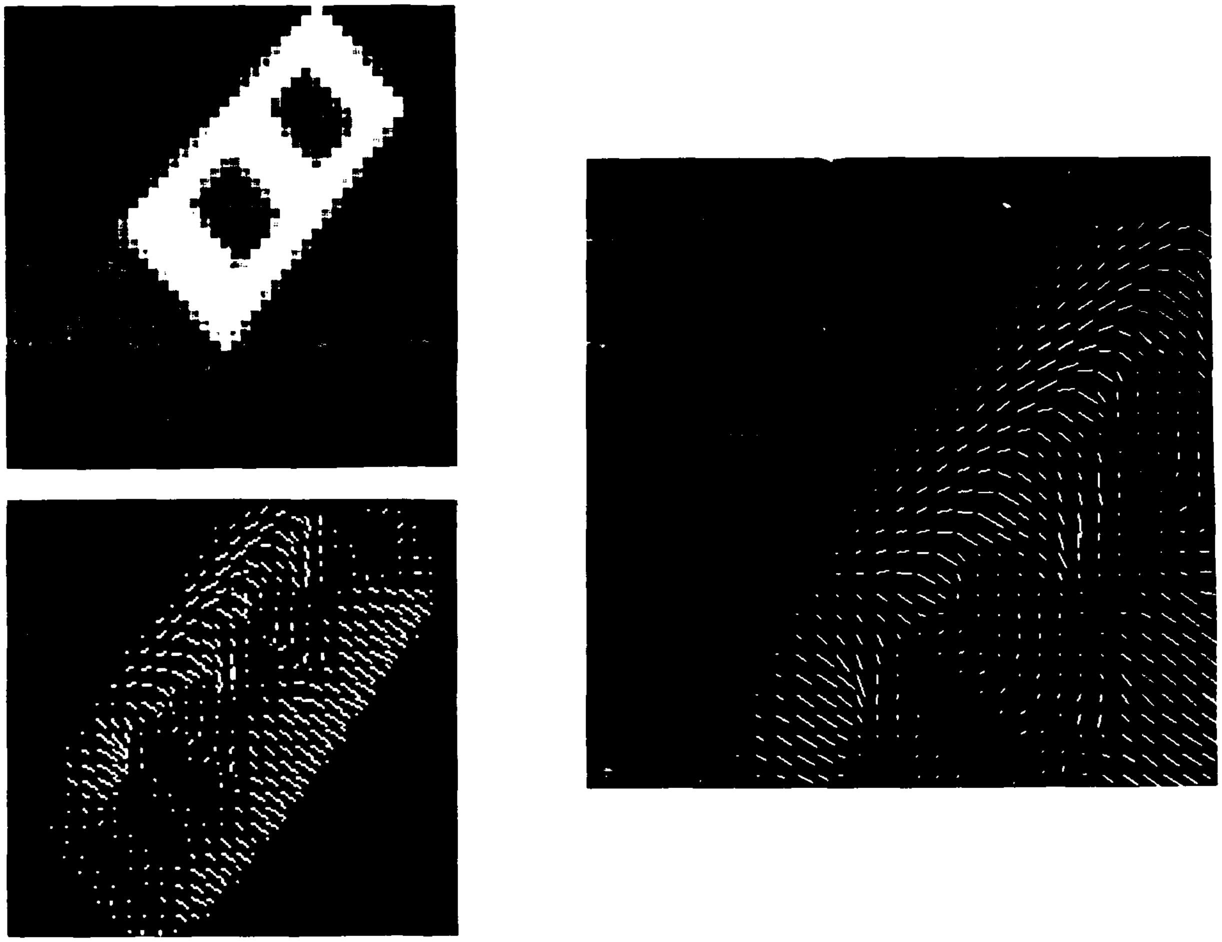


Figure 4: Results of Implementation

4 Conclusion

We have proposed information-fusion as an approach to robust estimation of image-flow. Two levels of fusion are suggested. We have developed a computational model to incorporate low-level fusion in a matching based approach. Our model is computationally local in nature. The underlying paradigm in our approach is to derive several linear constraints on the actual image-velocity along with the associated confidence measures and then iteratively find a velocity field that minimizes the error in these constraints. The constraints can be classified into two categories, (i) convection constraints and (ii) neighborhood constraints. Convection constraints are derived only once at the onset by explicit matching across the images. Neighborhood constraints are derived at each iteration from the distribution of the flow-field (from the previous iteration) in a small neighborhood of the point under consideration. Currently, an apriori estimate of the maximum possible image-displacement is required in order to establish the size of the search window. However, this restriction can be relaxed by using a hierarchical approach similar to that of Anandan [3]. Results of implementation of our model on real imagery are appear very promising. A detailed error analysis is currently in progress.

Although we have used a matching based approach to illustrate the underlying principle of our model, it is general enough to be used with a gradient-based approach. All that has to be changed is the derivation of the *convection constraints lines*. The derivation of the *neighborhood constraints lines* stays unchanged. The derivation of the convection constraint lines for a gradient based approach is quite straightforward under the assumption of a locally translational motion. The directional gradients of the image intensity in x and y directions can be assumed to be considered separately to derive two linear constraints. In this case, the directional gradients themselves can be used as measures of the confidences in the respective directions. Furthermore, our model offers a framework to integrate gradient-based and matching-based methods to achieve high-level fusion. This could be done by collecting all the constraints from gradient measurements and matching measurements and then using some a-priori estimate of the relative reliability to simultaneously satisfy all the constraints. This possibility is currently under investigation.

Acknowledgements: This research was funded in part by DARPA grant N00039-84-C-0165. I would like to thank Peter Allen, Terry Boult, Leo Dorst, Michael Shneier and Thom Warmerdam for their useful advice during the course of this work and Bill Yoshimi for help with implementation.

References

[1] E.H. Adelson and J.R. Bergen. Spatiotemporal energy models for the perception of motion. *Journal of the Optical Society of America*, 2(2):284–299, 1985.

[2] J.K. Aggarwal and N. Nandhakumar. On the computation of motion from sequences of images - a review. Technical Report TR-88-2-47, Computer Vision Research Center, University of Texas at Austin, 1988.

[3] P. Anandan. Measuring visual motion from image sequences. Technical Report COINS-TR-87-21, COINS Department, University of Massachusetts, Amherst, 1987.

[4] D. Ballard and C. Brown. *Computer Vision.* Prentice Hall, Inc., Englewood Cliffs, NJ, 1982.

[5] M.L. Braunstein. *Depth Perception through Motion.* Academic Press, New York, 1976.

[6] P.J. Burt, C. Yen, and X. Xu. Multi-resolution flow-through motion analysis. In *Proceedings of the IEEE CVPR Conference*, pages 246–252, 1983.

[7] B.F. Buxton and H. Buxton. Computation of optic flow from the motion of edge features in image sequences. *Image and Vision Computing*, 2, 1984.

[8] D. Heeger. A model for extraction of image flow. In *First International Conference on Computer Vision*, 1987.

[9] E.C. Hildreth. *The Measurement of Visual Motion.* MIT Press, 1983.

[10] B.K.P Horn and B. Schunk. Determining optical flow. *Artificial Intelligence*, 17:185–203, 1981.

[11] H.H. Nagel. On the estimation of dense displacement vector fields from image sequences. In *Proceedings of the Workshop on motion: Representation and Perception, Toronto*, pages 59–65, 1983.

[12] K. Prazdny. On the information in optical flows. *Computer Vision, Graphics and Image Processing*, 22:239–259, 1983.

[13] G.L. Scott. Four-line method of locally estimating optic flow. *Image and Vision Computing*, 5(2), 1986.

[14] A. Singh. Measurement of visual motion. Technical Report CUCS-287-87, Department of Computer Science, Columbia University, New York, 1987.

[15] A. Singh. Aperture problem: A review and a new local solution. Technical Report TR-88-082, Philips Laboratories, Briarcliff Manor, New York, 1988.

[16] M. Subbarao. Interpretation of image motion fields:rigid curved surfaces in motion. Technical Report CS-TR-1654, Computer Science Department, University of Marylnd, 1986.

[17] H. Wallach and D.N. O'Connell. The kinetic depth effect. *Journal of Experimental Psychology*, 45:205–207, 1953.

[18] A.M. Waxman and K. Wohn. Contour evolution, neighbourhood deformation and global image flow. Technical Report CAR-TR-58, Center for Automation Research, University of Maryland, 1984.

[19] M. Yachida. Determining velocity maps by spatiotemporal neighbourhoods from image sequences. *CGIP-21*, pages 262–279, 1983.

Description and Interpretation of Rotational Motion from Image Trajectories

Harpreet S. Sawhney John Oliensis

Computer and Information Science Department
University of Massachusetts
Amherst, MA 01003

Abstract

We propose that constructing global spatial organisations from individual token trajectories is a powerful technique for motion and structure interpretation. Recovering 3D motion from grouped trajectories has many analogies with 3D structure-from-contours in static imagery, which have not been made explicit in the motion vision literature. Our proposal is also similar to Stevens' [17] idea of strongly suggested geometric groupings in Glass patterns. Additionally, for our motion model, the grouped trajectories exploit the symmetry inherent in the motion. We demonstrate our approach through the solution of scene motion and structure parameters for a set of 3D points rotating rigidly around an arbitrary axis. Instead of using individual point correspondences directly, we fit conic section curves to grouped sets of point correspondences and compute, in closed form, the 3D motion and structure parameters using the quadratic form matrix representing each conic. We analyze ambiguities in the solutions obtained from the image path of a single point and arrive at a unique solution using multiple point traces.

1. Introduction

Most motion algorithms make few presuppositions about the environment, and rely on the relatively unstructured computation of point displacements in 2 or 3 frames of image data. We propose here instead that coherent motion percepts can be derived from global geometric structures in the image motion. Specifically, we consider the recovery of motion and structure for a single, rigidly rotating object using image trajectories. We will demonstrate that individual image point displacements provide only weak constraints on the corresponding 3D trajectories. We therefore argue for the necessity of grouping individual image trajectories into global curves exploiting the symmetry of the motion. From these curves the 3D motion and structure can be accurately represented and recovered.

First, a method for reconstructing in closed form the 3D trajectory of a point rotating rigidly around a fixed axis given its image path is described. For this method to succeed, a reliable estimate of a substantial portion of the full 360 degree image path is needed, since the accuracy of the trajectory fit degrades drastically for small segments. The structure and motion of a rotating object can be completely recovered in this way, up to a single unavoidable scale ambiguity, if many points on the object are tracked. Although we do not have a grouping algorithm, we show that manual linking of individual tracked displacements yields estimates for the images trajectories of the required accuracy. Our linking makes explicit the symmetry of the 3D motion and structure.

For small rotations, motion and structure recovery on the basis of short individual point tracks fails. We emphasize that this failure is intrinsic, not an artifact of a particular fitting routine, and that it provides a compelling argument for the necessity of grouping in motion interpretation. There is also an interesting analogy with the psychophysical literature on deriving global percepts from geometrical organization. Our proposal is similar to Stevens' [17] idea of strongly suggested geometric groupings in Glass patterns. In the psychophysical motion literature, our work is related to Todd's [18] interesting study of perception of global regularities in rigid and non-rigid motion. Motion interpretation from point displacements grouped into motion contours also has obvious analogies with 3D structure-from-contours in static imagery—for instance computing the FOE is comparable to determining the vanishing point in static images.

Although it is particularly easy to envisage the appropriate global organizations for simple motions like pure translation and rotation, the power of the idea that coherent motion percepts arise from global structure seems general and compelling.

2. Comparison To Related Work

The standard two frame approach to the computation of 3D motion and structure requires minimal assumptions, but does not permit natural descriptions of motions. For instance, the motion of a purely rotating soccer ball will typically be described in terms of a rotation around an axis passing through the camera center rather than through that of the ball. It can be argued that the second, natural description can be constructed from a series of two-frame rotation and translation computations, but the robustness of this is questionable when each computed pair is itself obtained through a non-linear constrained optimization. For a comprehensive survey of many of the two frame approaches see Barron [4].

Another problem with two frame approaches is that structure computation is based only on one displacement measurement for each feature, so that depth recovery may be erroneous even for correctly determined motion parameters. This problem is ameliorated for known motion, or motion with a stereo camera arrangement, since then the two frame depth computation can be incrementally refined [10].

One can also hope to overcome these problems by explicitly posing the 3D interpretation problem as a multi-frame problem, but this typically requires constraints on the motion, for instance that motion parameters remain constant over a period of time. Weng et al. [20], Chellapa et al. [23], Shariat [16] and Pavlin [13] use various forms of motion constancy over time and solve for the underlying 3D motion and/or structure parameters using a variety of non-linear parameter estimation techniques. Webb and Aggarwal [19] solve for the parameters of rotational motion using ellipses, but they are limited to *orthographic* projection only. (See also the very recent paper by Jaenicke [8].) Our approach is similar to those above but potentially applicable to less-constrained motions, since it makes use of spatial information to supplement temporal correspondences, and thus requires fewer frames and a shorter period of motion constancy. Moreover, this approach may be an important first step not only for 3D parameter estimation but also for detecting occlusions/deocclusions and object segmentation. A related approach to detecting occlusions is that of Baker et al. [3]. For a survey of various multi-frame approaches see Aggarwal et al. [1].

At the outset, we caution that the cited works treat more general models of motion than the one considered here. However, we hope to generalize it beyond the demonstration model studied in this paper. It has implications not only for a novel approach to 3D motion and structure computation, but also for a unified view of 2D grouping and 3D interpretation.

An innovation of our approach is that we do not compute motion directly from individual feature correspondences, but rather from extended image trajectories. Sethi et al. [15] also discuss image trajectories but do not use them to find 3D parameters. In addition, we show how linking local image trajectories into global spatial organizations can lead to good qualitative and quantitative motion percepts. For instance, Fig. 1 depicts imaged points on two spheres rotating around different axes. The point tracks immediately suggest geometric organizations (elliptic arcs) similar to Glass patterns (see Stevens [17]). For rotations, each track of a point is a conic section curve. Our approach envisages grouping individual point tracks on the basis of smoothly varying relations among the 2D parameters describing these tracks, i.e. based purely on image plane characteristics. The result can be exploited to provide powerful constraints on a globally coherent motion, as demonstrated in this work. We do not have an algorithm yet for capturing these groupings but our demonstration makes a compelling case for further research which we are carrying out. More complex motions and grouping relations are under study.

3. The Problem: A Rotating Rigid Body in Space

We consider the case of a sequence of images of a rotating rigid body in space or the camera rotating around an arbitrary axis with respect to a fixed environment. Image point correspondences are assumed given over a period of time, and the camera parameters are known. The problem to be solved is that of determining the orientation and location of the rotation axis, and, for each point on the body corresponding to an identified image point sequcnce, the location of its 3D trajectory. We assume perspective projection, and, for simplicity, a square image.

A set of parameters sufficient to define the problem geometry is depicted in Fig. 2 :

$\hat{b}$: *Unit vector in the direction of the rotation axis.*

$\vec{c}$: *Location vector of the rotation axis, given by the point where the axis*
intersects a plane normal to it that passes through the origin.

$\vec{r}$: (x, y, z), *Location vector of a 3D point on the body.*

d : *Location of the center of the circular 3D trajectory of a body point, given by its signed*
distance from the point $\vec{c}$ *and measured positive in the* *positive* $z-$ *direction.*

k : *Radius of a circular 3D trajectory,* $k > 0$.

f : *Focal length of the camera.*

$\vec{R}$: (X, Y, f), *image vector in homogeneous pixel coordinates.*

This parameterization allows an easy separation of motion from structure parameters. We represent a vector as $\vec{v}$, a unit vector as $\hat{v}$ and $\boldsymbol{v}$ as the corresponding column vector. Quantities enclosed in square brackets , e.g. $[M]$, represent matrices.

The rotation axis can be specified using a minimum of four parameters [14]. Two additional parameters specify each circular trajectory of a body point relative to the rotation axis. These unknowns can be determined from the images only up to an arbitrary scale factor, the knowledge of which fixes the values of all parameters. Thus, for n 3D trajectories, there are $2n + 3$ determinable unknowns. Each image point in each frame gives one constraint equation assuming that the motion does correspond to the model under consideration, namely rigid rotational motion. Therefore one 3D point imaged in five frames, two 3D points imaged in four frames, or more than two 3D points all imaged in more than two frames, provide adequate information for a solution. In order to provide robustness in the face of noise in image measurements and small model deviations, however, more information (i.e. more frames) is necessary.

4. Formulation

An outline of our approach to the problem described above is as follows :

- We write down an expression giving explicitly the perspective projection onto the image plane of a circular trajectory in space. This projection is a general conic section, either a straight line segment, circle, ellipse, parabola, or hyperbola.
- Conditions determining the type of the conic section of the projection are specified.
- Using a general conic fit algorithm, the sequence of image points identified with the motion of a single body point is fit to a conic section. This requires at least five frames for each body point. The conic section is represented as a quadratic form in the homogeneous image coordinates.
- The 3D trajectory of the point is then solved for in closed form in terms of the conic section fit to the image data, up to an apparent eight-fold ambiguity.
- Six of the eight possible solutions can be rejected by invoking the scene-in-front-of-image criterion, or because they duplicate physically the other solutions.
- The remaining two-fold ambiguity can be resolved by requiring that different 3D trajectories share the same axis of rotation.
- Finally, we obtain a best-fit for the axis of rotation by combining information from all 3D points.

We first derive an expression for the image plane projection of a circular trajectory in space parameterized as depicted in Fig. 2. It is evident that our parameterization obeys the following conditions :

$$((\vec{r_i} - \vec{c}) - d_i\hat{b}) \cdot ((\vec{r_i} - \vec{c}) - d_i\hat{b}) = k_i^2 \tag{1}$$

$$\vec{r_i} \cdot \hat{b} = d_i \tag{2}$$

$$\vec{c}.\hat{b} = 0 \qquad \hat{b}.\hat{b} = 1 \tag{3}$$

The index i specifies the particular 3D point on the rigid body; we will drop it in the following treatment where there is no ambiguity in doing so.

The perspective equation in homogeneous coordinates is :

$$\vec{R} = f\frac{\vec{r}}{\vec{r}.\hat{z}} \tag{4}$$

From Equation (2) , using similar triangles, one obtains :

$$\vec{r} = \frac{d}{\vec{R} \cdot \hat{b}}\vec{R} \tag{5}$$

(The degenerate case where $d = \vec{R}\cdot\hat{b} = 0$ causes no difficulty and is discussed later.) Substituting Equation (5) into Equation (1) and rearranging terms, we get :

$$\frac{d^2}{(\vec{R}\cdot\hat{b})^2}\vec{R}\cdot\vec{R} - 2d\frac{\vec{R}\cdot\vec{c}}{\vec{R}\cdot\hat{b}} + \vec{c}\cdot\vec{c} - d^2 - k^2 = 0 \tag{6}$$

After multiplication through by $(\vec{R}\cdot\hat{b})^2$, this can be rewritten in terms of a symmetric quadratic form matrix. In boldface notation :

$$\boldsymbol{R}^T[\ d^2[I] - d[\ \boldsymbol{c}\boldsymbol{b}^T + \boldsymbol{b}\boldsymbol{c}^T\ ] + (\boldsymbol{c}^T\boldsymbol{c} - d^2 - k^2)[\boldsymbol{b}\boldsymbol{b}^T]\ ]\boldsymbol{R} = 0 \tag{7}$$

This equation represents a general conic in X and Y, the image plane coordinates. The linear and constant terms in the standard form of a conic [12] are included in this expression since $\vec{R}$ is in homogeneous form and includes a constant term. Thus, given a circle as the 3D trajectory of a point, the expected image projection is determined by the matrix :

$$[M_{exp}] = [\ d^2[I] - d[\boldsymbol{c}\boldsymbol{b}^T + \boldsymbol{b}\boldsymbol{c}^T] + (\boldsymbol{c}^T\boldsymbol{c} - d^2 - k^2)[\boldsymbol{b}\boldsymbol{b}^T]\ ] \tag{8}$$

The image will be an ellipse if the whole of the circle incorporating the 3D trajectory lies on the positive z side of the xy plane, i.e. if all points on this circle have positive z component. The image will be a hyperbolic arc when the 3D circle intersects the xy-plane in exactly two points. The four possible directions of approach to these intersections determine the four asymptotes in the image plane. Finally, when the 3D circle is tangent to the xy-plane, the imaged arc is a section of a parabola. Again, the two possible directions of approach towards the tangent point generate the two paths which become unbounded in the image. In the latter two cases, the image trace—i.e the image projection of a 3D trajectory—is not closed. In the first case of an ellipse, the image trace may be either a closed curve—a complete ellipse—or an open partial ellipse.

Equation (7) specifies the imaged conic section corresponding to a 3D circle. Below, we show how starting from a conic section in the image plane, one can recover the 3D circle of which it is the projection, up to a scale ambiguity. Thus, given our model of rigid rotational motion, if it is possible to track a particular body point through sufficiently many image frames to reliably fit its image trajectory to a conic section, its 3D circular trajectory can also be determined. In a realistic scenario where only a small part of a single point's trajectory may be available, the conic section fits tend to be locally accurate but globally erroneous. However, we will show that by doing spatio-temporal groupings across multiple point traces, good globally correct image conics can be described and hence fairly correct 3D parameters extracted.

5. Solution

An arbitrary conic section in the image plane can be specified by a quadratic form as in Equation (7), with a symmetric 3-by-3 matrix $[M_{com}]$. The matrix $[M_{com}]$ is derived from the image data by a best fit. The corresponding 3D circle is determined by computing the values of the 3D parameters d, k, $\vec{c}$, $\hat{b}$ that yield a matrix $[M_{exp}]$ specifying the same conic section as $[M_{com}]$. Note that any scalar multiple of the matrix $[M_{exp}]$ specifies the same image curve, so that the 3D parameters can only be recovered up to a multiplicative ambiguity. This is the scale ambiguity mentioned earlier: only the ratios d_n, k_n, c_n and $\hat{b}$ are recoverable, where

$$d_n = \frac{d}{|\vec{c}\,|} \qquad k_n = \frac{k}{|\vec{c}\,|} \qquad \vec{c_n} = \frac{\vec{c}}{|\vec{c}\,|} \qquad \boldsymbol{c}_n^T\boldsymbol{c}_n = 1 \tag{9}$$

(We are assuming here that the rotation axis does not pass through the origin, i.e. that $\vec{c}$ is not the zero vector. This case is easily distinguished from the image data, and will be treated separately.)

Since the absolute magnitudes of the 3D parameters can not be determined, we rescale $[M_{exp}]$ by the magnitude squared of the location vector $\vec{c}$. The rescaled matrix can be written in terms of the recoverable ratios :

$$[M_{exp}] = [\ d_n^2[I] - d_n[\boldsymbol{c}_n\boldsymbol{b}^T + \boldsymbol{b}\boldsymbol{c}_n^T] + (1 - d_n^2 - k_n^2)[\boldsymbol{b}\boldsymbol{b}^T]\ ] \tag{10}$$

The ratios are computed from the image data by requiring that $[M_{exp}]$ is equal to $[M_{com}]$ up to a scale factor, which can also be found. We use a two-stage conic fit algorithm to determine $[M_{com}]$. Bookstein's [6] algorithm, which uses an algebraic distance measure, is used first to get a closed form conic fit solution to the set of point correspondences corresponding to a body point. This is inputted as an initial guess to an iterative quasi-Newton algorithm, which uses an approximation to the first-order euclidean distance from the conic as an error measure. In case of noisy correspondences and small deviations from modeled motion, many more than five frames are necessary to get a good conic fit. For each identified body point, the resulting best fit conic section to the path of its image projections is specified and represented by a matrix $[M_{com}]$.

5.1 Axis Through Origin

We first consider the simpler case where the rotation axis passes through the origin. Then the expected conic form matrix is :

$$[M_{exp}] = [\ d^2[I] - (d^2 + k^2)[\boldsymbol{b}\boldsymbol{b}^T]\ ] \tag{11}$$

As above, we normalize this by d, assuming it is non-zero, and write the matrix in terms of the ratio k/d which is recoverable. (The solution for the special case when d is zero follows trivially from the non-zero case.) The normalized matrix is :

$$[M_{exp}] = \alpha[\ [I] - (1 + \frac{k^2}{d^2})[\boldsymbol{b}\boldsymbol{b}^T]\ ] \tag{12}$$

We have included a scale factor α to represent explicitly the multiplicative ambiguity discussd above.

The eigenvalue-eigenvector pairs for this matrix are :

$$\lambda_1 = -\alpha(\frac{k}{d})^2 \quad \lambda_2 = \alpha \quad \lambda_3 = \alpha \qquad \boldsymbol{n}_1 = \boldsymbol{b} \quad \boldsymbol{n}_2 = \boldsymbol{n}_2 \quad \boldsymbol{n}_3 = \boldsymbol{n}_3 \tag{13}$$

where n_2 and n_3 are any two independent vectors in a plane normal to $\boldsymbol{b}$. Note that two eigenvalues are identical and the third one is of a different sign and magnitude. In the more general case where the rotation axis does not intersect the origin this redundancy of the eigenvalues will not occur.

In order to recover the 3D parameters of the trajectory from the image data, we compute the eigenvalues of the matrix $[M_{com}]$ derived from the conic fit algorithm. If two of the eigenvalues are same and third one is different in magnitude and of opposite sign, this implies that the sequence derives from a point rotating around an axis passing through the origin. Let the eigenvalues of $[M_{com}]$ be λ_1, and $\lambda_2 = \lambda_3$. Since $[M_{com}] = [M_{exp}]$ (after adjustment of the scale factor α), these eigenvalues can be identified with the eigenvalues of $[M_{exp}]$ calculated above. Then,

$$\alpha = \lambda_2 \ (\ or\ \lambda_3) \qquad \frac{k}{d} = \pm\sqrt{-\frac{\lambda_1}{\alpha}} \tag{14}$$

and $\boldsymbol{b}$ is the unit eigenvector for λ_1. Without loss of generality, $\boldsymbol{b}$ can be forced to lie in the hemisphere of positive z directions. Then a unique sign for $\frac{k}{d}$ can be determined by invoking the fact that the imaged 3D trajectory must lie in front of the camera. For the more general case, the resolution of this type of sign ambiguity will be discussed at length.

Hence, for this case there is a unique, closed-form solution for the circular trajectory in space given $[M_{com}]$, the conic section fit to the image point sequence.

5.2 Generic Case: Rotation Axis Not Through Origin

As in the special case above, we determine the 3D trajectory of a body point by identifying the eigenvalues of $[M_{exp}]$ computed in terms of the 3D parameters with those of the image-derived matrix $[M_{com}]$. For convenience, we rewrite $[M_{exp}]$ with an explicit factor α as in the previous section and drop the subscript indicating ratios :

$$[M_{exp}] = \alpha[\ d^2[I] - d[\boldsymbol{c}\boldsymbol{b}^T + \boldsymbol{b}\boldsymbol{c}^T] + (1 - d^2 - k^2)[\boldsymbol{b}\boldsymbol{b}^T]\ ] \tag{15}$$

We first derive the eigenvalues of this matrix. As $\boldsymbol{b}$ and $\mathbf{c}$ are orthogonal, a rotated coordinate system can be chosen in which :

$$\boldsymbol{b} = [\ 0\ 0\ 1\ ] \qquad \boldsymbol{c} = [\ 0\ 1\ 0\ ] \tag{16}$$

It is a standard result of linear algebra that the eigenvalues of a matrix remain the same and can be computed in an arbitrary rotated coordinate system. After substituting Equation (16) into Equation (15), $[M_{exp}]$ has the simple form in this rotated system :

$$M'_{exp} = \alpha \begin{bmatrix} d^2 & 0 & 0 \\ 0 & d^2 & -d \\ 0 & -d & 1 - k^2 \end{bmatrix}$$

The three eigenvalues of this matrix are :

$$\lambda_1 \;=\; \alpha\, d^2 \tag{17}$$

$$\lambda_2 \;=\; \frac{\alpha}{2}\,\left(\,(1+d^2-k^2)+\sqrt{(1+d^2-k^2)^2+4d^2k^2}\,\right) \tag{18}$$

$$\lambda_3 \;=\; \frac{\alpha}{2}\,\left(\,(1+d^2-k^2)-\sqrt{(1+d^2-k^2)^2+4d^2k^2}\,\right) \tag{19}$$

For α positive, λ_1 and λ_2 are positive and λ_3 is negative, except in the degenerate case when $d\;=\;0$, corresponding to an image trajectory that is a degenerate conic, i.e. a straight line segment. We always choose to have α positive, and thus, for consistency, normalize the computed matrix $[M_{com}]$ similarly so that only one of its eigenvalues is negative and two are positive. Hence the negative eigenvalue of $[M_{com}]$ can be uniquely identified with λ_3. Also, the larger of the two positive eigenvalues of $[M_{com}]$ can be uniquely identified with λ_2, which one can show is always larger than λ_1 except possibly in the degenerate case.

The three eigenvalues of $[M_{com}]$ can therefore be assigned unambiguously to λ_1, λ_2 and λ_3, corresponding to Equations (17), (18) and (19), respectively, and the 3D parameters d, k and α can be solved for in terms of these eigenvalues. Let $\gamma_1 \equiv \lambda_2/\lambda_1$ and $\gamma_2 \equiv \lambda_3/\lambda_1$. Then :

$$d^2=\frac{1}{\gamma_1+\gamma_2-\gamma_1\gamma_2-1} \qquad k^2=-\gamma_1\gamma_2 d^2 \qquad \alpha=\frac{\lambda_1}{d^2} \tag{20}$$

Thus d and k are specified up to a sign ambiguity in d. We discuss the sign ambiguities of our solution in the next section.

$\boldsymbol{b}$ and $\boldsymbol{c}$ can also be obtained as follows. It is evident from Equation (15) that one of the eigenvectors of $[M_{exp}]$ is a vector $\boldsymbol{n}_1$ normal to the plane formed by $\boldsymbol{b}$ and $\boldsymbol{c}$. Since $\boldsymbol{n}_1$ satisfies :

$$[M_{exp}]\boldsymbol{n}_1 \;=\; \alpha\, d^2\boldsymbol{n}_1 \;=\; \lambda_1\boldsymbol{n}_1 \tag{21}$$

it is associated with the eigenvector λ_1 . The other two eigenvectors $\boldsymbol{n}_2$ and $\boldsymbol{n}_3$ with associated eigenvalues λ_2 and λ_3, respectively, *must span the plane formed by* $\boldsymbol{b}$ and $\boldsymbol{c}$, since all the eigenvectors are mutually orthogonal. Therefore :

$$\hat{c}=\cos\theta\;\hat{n}_2+\sin\theta\;\hat{n}_3 \qquad \hat{b}=\sin\theta\;\hat{n}_2-\cos\theta\;\hat{n}_3 \tag{22}$$

Further,

$$[M_{com}]\boldsymbol{c} \;=\; [M_{exp}]\boldsymbol{c} \;=\; \alpha\, d^2\boldsymbol{c}-\alpha\, d\,\boldsymbol{b} \;=\; \lambda_2\cos\theta\;\boldsymbol{n}_2+\lambda_3\sin\theta\;\boldsymbol{n}_3 \tag{23}$$

$$[M_{com}]\boldsymbol{b} \;=\; [M_{exp}]\boldsymbol{b} \;=\; -\,\alpha\, d\;\boldsymbol{c}+\alpha(1-k^2)\;\boldsymbol{b} \;=\; \lambda_2\sin\theta\;\boldsymbol{n}_2-\lambda_3\cos\theta\;\boldsymbol{n}_3 \tag{24}$$

Substituting Equation (22) into (23) and (24), one obtains :

$$\tan\theta=\frac{\alpha\,d^2-\lambda_2}{\alpha\,d}=\frac{\alpha\,d}{\lambda_3-\alpha\,d^2}=\frac{\alpha\,d}{\alpha(1-k^2)-\lambda_2}=\frac{\lambda_3-\alpha(1-k^2)}{\alpha\,d} \tag{25}$$

Thus, $\tan\theta$ can be computed in closed form in terms of the image parameters up to the sign ambiguity in d. It follows that $\boldsymbol{b}$ and $\boldsymbol{c}$ can also be obtained in closed form up to sign ambiguities, by solving for the eigenvectors $\boldsymbol{n}_2$ and $\boldsymbol{n}_3$ of the image conic form matrix, which are identified unambiguously by their respective eigenvalues.

Hence, apart from the sign ambiguities, all the 3D parameters of the trajectory can be uniquely computed in terms of the eigenvalues and eigenvectors of $[M_{com}]$, the computed image conic form matrix.

6. Multiple Solutions

There are two solutions for d in Equation (20). For each of these, there are four solutions for $\boldsymbol{b}$ and $\boldsymbol{c}$ from the four sets of signed values of $\boldsymbol{n}_2$ and $\boldsymbol{n}_3$ in Equation (22). These *eight* solutions can be written as two sets of four, each corresponding to the same k :

$$S_1 \;=\; \{\;\{\boldsymbol{b}_1,\boldsymbol{c}_1,d_1\},\;\;\{-\boldsymbol{b}_1,-\boldsymbol{c}_1,d_1\},\;\;\{\boldsymbol{b}_1,-\boldsymbol{c}_1,-d_1\},\;\;\{-\boldsymbol{b}_1,\boldsymbol{c}_1,-d_1\}\;\}$$

$$S_2 \;=\; \{\;\{\boldsymbol{b}_2,\boldsymbol{c}_2,d_1\},\;\;\{-\boldsymbol{b}_2,-\boldsymbol{c}_2,d_1\},\;\;\{\boldsymbol{b}_2,-\boldsymbol{c}_2,-d_1\},\;\;\{-\boldsymbol{b}_2,\boldsymbol{c}_2,-d_1\}\;\}$$

The ambiguity within each set of four is evident, since the different signed values for the parameters all lead to the same computed conic form matrix in Equation (15). To see the relation between the two sets of solutions S_1 and S_2, refer to Fig. 3. Reflecting each of $\{b_i, c_i\}$ in S_1 across any one of the eigenvectors, n_2 or n_3, leads to a corresponding solution in the other set. Fig. 3 depicts one of these reflections across n_3. The result of this reflection again yields a conic form matrix with the same sets of eigenvalues and eigenvectors. Hence, given the measured image conic form matrix, these eight indeed are solutions. Within the constraints of Equation (3), these are the only possible solutions because they exhaust all options in the representation of the given conic form matrix in terms of its eigencomponents, which in turn is a complete representation.

The apparent eight-fold ambiguity found above is not real. Since $\{b, c, d\}$ and $\{-b, c, -d\}$ represent the same point along the rotation axis, four of the above solutions simply duplicate physically the other four. This ambiguity can be eliminated by our convention that the vector b always lies in the positive z–component hemisphere. This leaves two remaining solutions in each set. We next impose the constraint that a 3D point must lie in front of the image plane in order to be imaged. [7]. From Equation 5, this implies that $\beta = \frac{d}{\hat{R}.b}$ must be positive. If the parameter pair $\{b, d\}$ satisfies this constraint, then the alternate set $\{-b, d\}$ cannot. Thus, two more solutions are eliminated, one from each set. The remaining two solutions, one from each set, cannot be disambiguated from the image trace of one point alone. This is similar to the well known ambiguity of two-frame motion computation for a planar set of points (see [11]).

However, multiple image trajectories of 3D points rotating rigidly around a common axis can be used to resolve this ambiguity. The true solution for the axis will be appear as one of the possible solutions for *all* the points, while the second, incorrect solutions will be unique for each 3D point. The true solution is therefore easily picked out. The reason for the mismatch among the incorrect solutions is that, as illustrated in Fig. 4, the eigenvectors n_2 and n_3 of the matrix $[M_{com}]$ are different for each 3D point excepting when the axis b passes through the origin, which anyway can be handled differently as shown earlier. These vectors span the plane formed by the true b and c. Hence, the b and c solutions common to all 3D points make different angles with their respective spanning eigenvectors. As the second incorrect solutions are obtained by reflecting the true b and c across an eigenvector, clearly these solutions will be different for each 3D point. This is shown for two points in Fig. 4. Hence, the correct solution can be identified by combining information obtained from several image trajectories assuming they have already been segmented as a single rigid motion. In the presence of noise, the algorithm's ability to discriminate the true solution among the solutions for multiple trajectories will be limited by the similarity in their structure parameters.

7. Experiments: Grouping

We first describe our experiments with grouping. Fig. 5 and 6 show 256-by-256 image frames 1 and 10, respectively, of a chequered box which was rotated around an arbitrarily chosen fixed body axis using a cartesian robot arm. Rotation between each frame was approximately 4°. The object was imaged using a 16mm. focal length lens on a CCD camera at a distance of approximately 60cm. Points defined as intersections of a pair of lines were tracked over the sequence using the robust line-tracking system of Williams and Hanson [21]. This system incorporates the straight-line detection algorithm of Boldt and Weiss [5] and dense displacement fields computed between two successive frames using an algorithm by Anandan [2]. Detected straight lines are projected into successive frames using the displacement field, and a directed-acyclic-graph (DAG) of line correspondences over time is constructed. Line intersections are then tracked through this DAG representation. Figs. 7 and 8 show the sampled displacement field between frame 1 and 2 and a sample of tracked line segments, respectively.

In Fig. 9 are shown resulting sample sets of tracked points. Fig. 10 shows the result of describing a few individual tracks through fitted conics. Although the tracks represent nearly 80 degrees (20 frames) of the 3D circular arc, still the image conics fail to make explicit the common motion geometry across different point sets. However, the next figure, Fig. 11 shows conics fitted to manually linked point sets chosen because they are expected to lie along the same image trace. There is a dramatic improvement in the global percept of a common structure among these traces. The near-collinearity of their centers is strongly suggestive of a coaxial motion. Moreover, the motion and structure of these different points can now be quantitatively estimated with good accuracy. (The results are detailed in the next section.) We are in the process of researching algorithms to automatically capture these global compelling groupings for this as well as more general smooth motions.

We emphasize that the poor results obtained by fitting to individual tracks are not a failure of our fitting routine but inherent in the problem. Fitting an ellipse to a partial, slightly noisy elliptic arc is extremely sensitive to the noise. To human view, widely different ellipses are indistinguishably good fits along the arc itself. Moreover, the difficulties of individual fitting worsen as the viewing time is reduced and the tracked arcs become smaller. On the other hand, the grouping technique still leads to similar conic traces (Fig. 12), even if the time is reduced by half, that is to ten image frames. In this case, the individual curve descriptions are absolutely wrong vis-a-vis the desired global description although the point tracks fit very well the fitted curves. In fact, the best fits for most points were not even elliptic but hyperbolic.

An effective motion understanding system should successfully interpret even a rotation of only a few degrees of arc. In the extreme case, as Todd [18] suggests, human vision systems may be capable of this interpretation, based on the capture of compelling grouping relations, even from a two-frame span of data. Clearly, for such short viewing spans, it is difficult to interpret the motion from individual point traces. Even for the simplest case of pure rotation studied in this paper, it appears that spatial grouping is essential for the 3D interpretation of motion.

8. 3D Estimation Results

Results on both simulated and the box image data are reported here. In our simulation, we generated image point data for four 3D points rotating rigidly around an axis with direction ($\mathbf{b}$) $[1,1,1]$ and location ($\mathbf{c}$) $[7,-10,45]$. We imaged the four points for 50 frames with a 4 degree rotation between successive frames. A focal length of 160 pixels and image size of 256-by-256 were chosen for the camera model. We obtained four conic sections, one for each of the points, by fitting conic arcs to their discrete correspondences. Each of these was represented as a 3-by-3 conic form matrix M_{conn}. In one experiment, no noise was added to the image point coordinates while in the second, we added $(-1,1)$ pixel uniform noise before fitting the conics. We report the true and computed 3D parameters in Table 1 for both. In this table and the next, the true axis direction and location are given at the top. For each point, the two computed solutions for the rotation axis are shown in the body of the table at left, with the correct one on top. The correct solutions stand out as predicted since they are common to all points, in both the noisy and noise-free cases. At right are shown the true and computed normalized structure parameters. All computed parameters agree well with the actual ones.

For the box image, we show the results of estimating the axis location and orientation, and the structure parameters for four points. We use ellipses generated as fits to the linked image trajectories of symmetrically located points on the box as input to our solution method. The eight traces chosen are marked in Figs. 11 and 12. Table 2 shows the results for these eight traces. The ground truth for this image was obtained using a pose estimation algorithm developed by Kumar [9]. These estimated parameters are labelled as *true* in Table 2. Our results match well with the estimated ground truth. Here again, the wrong solution for each point is evident except for points 1, 4, 5 and 7 which are co-planar. Hence, for these the second solution too is the same.

9. Conclusions

We have presented and demonstrated a method for recovering in closed form the 3D structure and motion of a rigid body, based on fits to the image trajectories of selected body points assuming a particular model of motion, namely pure rotational motion. The paths traced by the 3D point projections in the image are first fit to conic sections, which are represented by quadratic form matrices. The eigenvalues and eigenvectors of these matrices are related to the parameters of the 3D trajectories, this relationship is inverted and the 3D trajectories solved for. Hence we obtain a complete recovery of the body structure and motion up to a single scale ambiguity. We gave a discussion and complete analysis of multiple solutions and their disambiguation, discussed the effects of noise, and described the results of experiments with real and synthetic images.

We also argued for the importance of global spatial groupings of individual token displacements in motion interpretation. For pure rotation, since full conic section curves are significantly underdetermined by any small sub-arcs, we argued that organizing trajectories into global curves can give improved results over simply combining the very uncertain motion interpretations based on short, individual point displacement trajectories. We also noted the relationship of this idea to psychophysical results on global organization, and proposed grouping as a general and powerful approach to motion interpretation.

Acknowledgments

We have derived constant intellectual and emotional support from Rakesh Kumar, Lance Williams, R. Manmatha, Brian Burns and P.Anandan throughout this work. We are grateful for Prof. Ed Riseman's consistent support and encouragement. Financial support for this research came from NSF/CER grant DCR8500332 and DARPA contracts F30602-87-C-0140.

REFERENCES

[1] J. K. Aggarwal and N. Nandhakumar. On the computation of motion from sequences of images - a review. Technical Report TR-88-2-47, University of Texas at Austin, 1988.

[2] P. Anandan. *Measuring Visual Motion from Image Sequences*. PhD thesis, University of Massachusetts at Amherst, 1987. COINS TR 87-21.

[3] H. H. Baker and R. C. Bolles. Generalizing epipolar-plane image analysis on the spatiotemporal surface. *Proceedings DARPA Image Understanding Workshop*, pages 1022–1030, 1988.

[4] J. Barron. A survey of approaches for determining optic flow, environmental layout and egomotion. Technical Report RBCV-TR-84-5, University of Toronto, 1984.

[5] M. Boldt and R.Weiss. Token-based extraction of straight lines. Technical Report COINS TR 88-, University of Massachusetts at Amherst, 1988.

[6] F. L. Bookstein. Fitting conic sections to scattered data. *Computer Graphics and Image Processing*, 9:56–71, 1979.

[7] B. K. P. Horn. Relative orientation. *Proceedings DARPA Image Understanding Workshop*, pages 826–837, 1988.

[8] R. A. Jaenicke. Structure from limited motion of complex objects. *Proceedings IEEE Wkshp. on Visual Motion*, pages 256–263, 1989.

[9] R. Kumar. Determination of camera location and orientation. *Proceedings DARPA Image Understanding Workshop*, 1989.

[10] L. Matthies and T. Kanade. The cycle of uncertainty and constraint in robot perception. *The Fourth International Symposium on Robotics Research*, pages 327–335, 1988.

[11] S. Negahdaripour and B. K. P. Horn. Direct passive navigation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 9(1):168–176, 1987.

[12] T. Pavlidis. Curve fitting with conic splines. *ACM Transactions on Graphics*, 2(1):1–31, 1983.

[13] I. Pavlin. Motion from a sequence of images. *Proceedings DARPA Image Understanding Workshop*, 1988. Cambridge, Massachusetts.

[14] K. S. Roberts. A new representation for a line. *Proceedings Computer Vision and Pattern Recognition*, pages 635–640, 1988.

[15] S. K. Sethi and R. Jain. Finding trajectories of feature points in a monocular image sequence. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 9(1), 1987.

[16] H. Shariat. *The Motion Problem: How to Use More than Two Frames*. PhD thesis, University of Southern California, Los Angeles, 1986.

[17] K. A. Stevens. Computation of locally parallel structure. *Biological Cybernetics*, 29:19–28, 1978.

[18] J. Todd. Visual information about rigid and non-rigid motion : A geometric analysis. *Journal of Experimental Psychology : Human Perception and Performance*, 8(2):238–252, 1982.

[19] J. A. Webb and J. K. Aggarwal. Structure from motion of rigid and jointed objects. *Artificial Intelligence*, 19:107–130, 1982.

[20] J. Weng, T. S. Huang, and N. Ahuja. 3-d motion estimation, understanding and prediction from noisy image sequences. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 9(3), 1987.

[21] L. R. Williams and A. R. Hanson. Translating optical flow into token matches. *Proceedings DARPA Image Understanding Workshop*, pages 970–980, April 1988.

[22] A. P. Witkin and J. M. Tenenbaum. On the role of structure in vision. *Human and Machine Vision 1983*, pages 481–543, 1983.

[23] G. Young and R. Chellappa. 3-d estimation using a sequence of noisy stereo images. *Proceedings Computer Vision and Pattern Recognition*, pages 710–716, 1988.

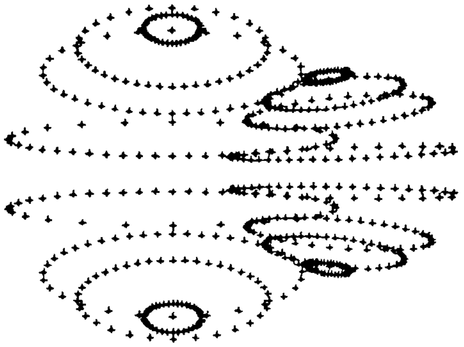


Fig. 1: Image point trajectories of two spheres rotating independently at different depths.

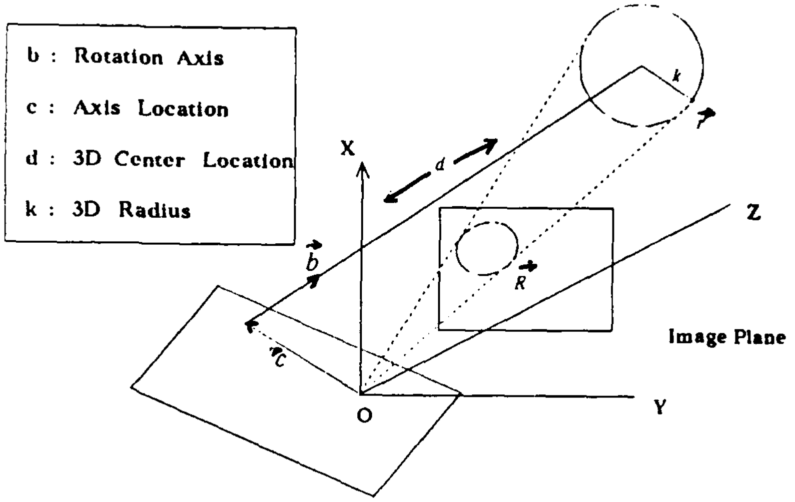


Fig. 2 : Geometry of a point's rotation

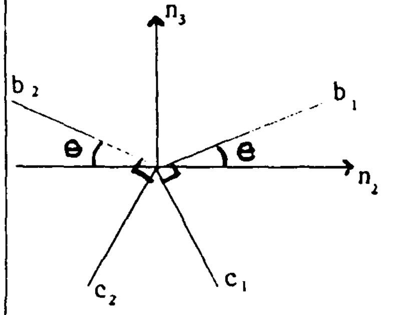


Fig. 3: Two final solutions for one point.

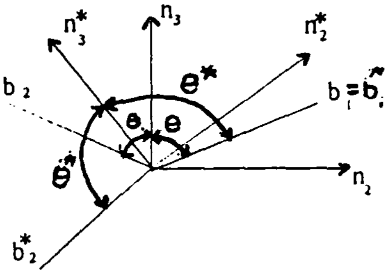


Fig. 4: Unique solution from two points.

Table 1: **Results for Four imaged 3D points rotating rigidly around an axis**

Data		**Axis dir.**			**Axis locn.**			**True**		**Computed**	
Type		b_x	b_y	b_z	c_x	c_y	c_z	**Structure**		**Structure**	
True		0.577	0.577	0.577	-0.603	-0.176	0.778	d	k	d	k
without noise in imaged data											
Pt 1	1	0.577	0.577	0.577	-0.603	-0.176	0.778	0.986	0.497	0.986	0.497
	2	-0.535	-0.111	0.837	0.640	0.593	0.488				
Pt 2	1	0.577	0.577	0.577	-0.603	-0.176	0.778	0.381	0.363	0.381	0.363
	2	-0.835	-0.525	0.168	0.004	0.298	0.955				
Pt 3	1	0.577	0.577	0.577	-0.603	-0.176	0.778	0.768	0.168	0.768	0.168
	2	-0.724	-0.310	0.616	0.415	0.518	0.748				
Pt 4	1	0.577	0.577	0.577	-0.603	-0.176	0.778	1.682	0.322	1.682	0.322
	2	-0.235	0.135	0.962	0.801	0.588	0.113				
with (-1,1) pixel uniform noise in imaged data											
Pt 1	1	0.570	0.577	0.585	-0.607	-0.183	0.773	0.986	0.497	1.003	0.502
	2	-0.529	-0.108	0.842	0.643	0.595	0.481				
Pt 2	1	0.579	0.579	0.573	-0.599	-0.174	0.781	0.381	0.363	0.378	0.363
	2	-0.834	-0.526	0.166	0.003	0.297	0.955				
Pt 3	1	0.582	0.590	0.560	-0.591	-0.166	0.790	0.768	0.168	0.736	0.174
	2	-0.730	-0.325	0.602	0.393	0.520	0.758				
Pt 4	1	0.583	0.577	0.572	-0.600	-0.169	0.782	1.682	0.322	1.651	0.317
	2	-0.243	0.132	0.961	0.800	0.587	0.121				

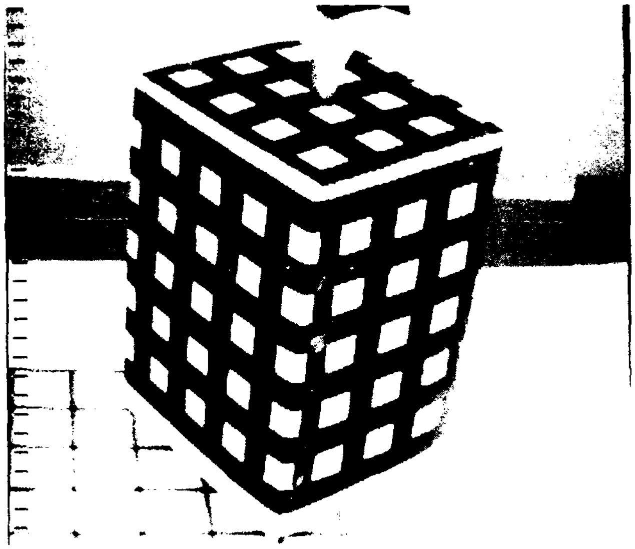


Fig. 5: Frame 1 of the box sequence

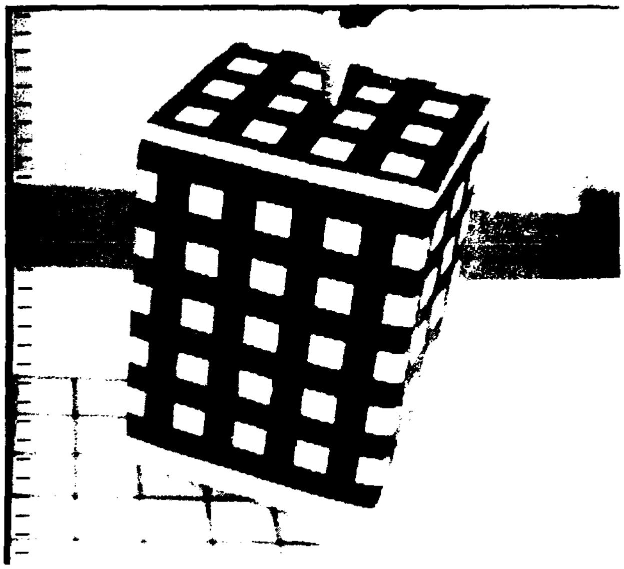


Fig. 6: Frame10 of the box sequence

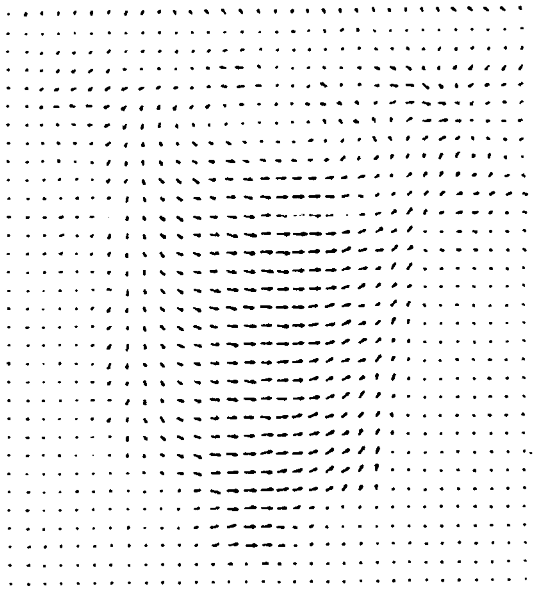


Fig. 7: Sub-sampled displacement field between frames 1 and 2

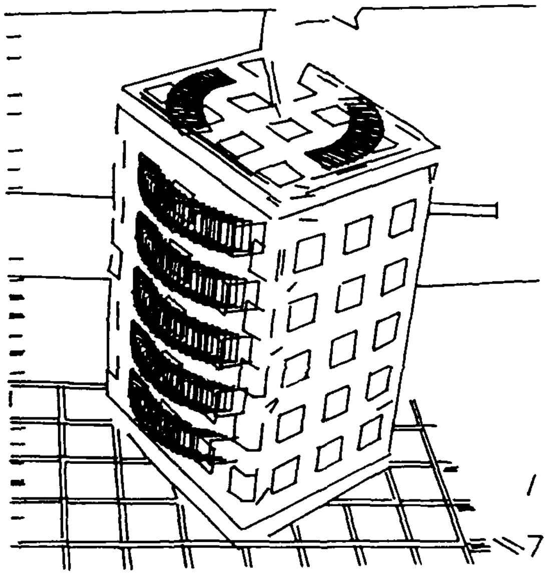


Fig. 8: Sample set of tracked lines

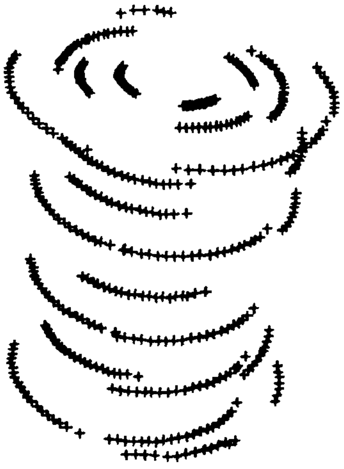


Fig. 9: Sample set of tracked line intersections

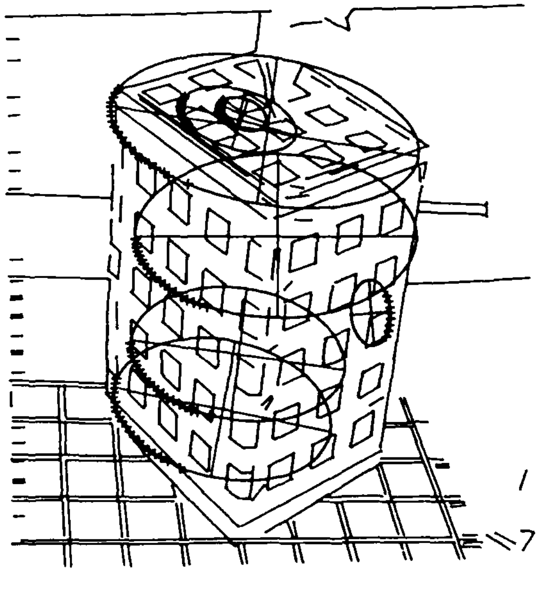


Fig. 10: Conic fits to **individual 20-frame** point tracks

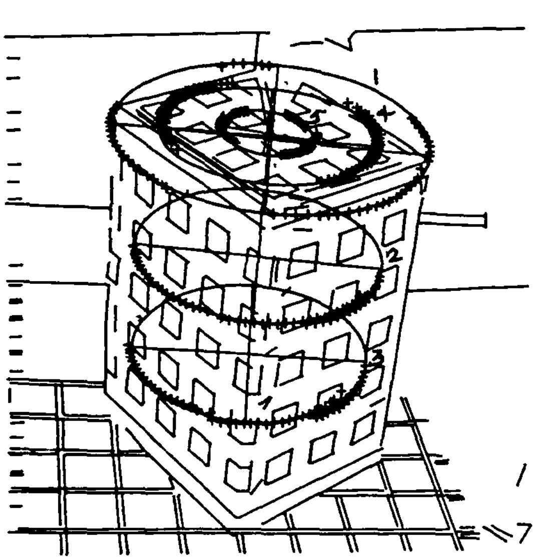


Fig. 11: Conic fits to set 1 of linked 20-frame point tracks

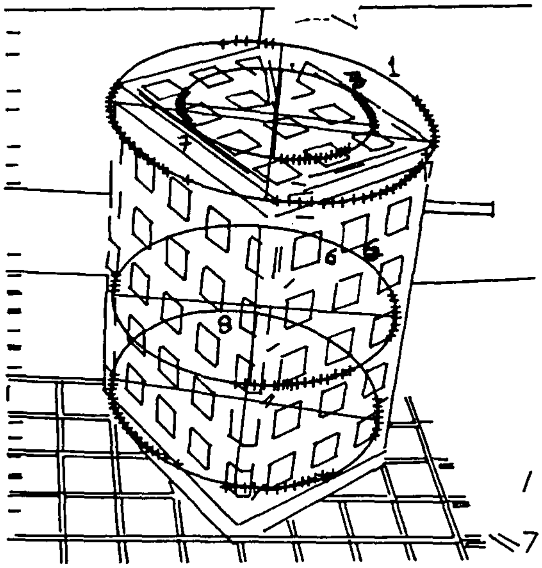


Fig. 12: Conic fits to set 2 of linked 20-frame point tracks

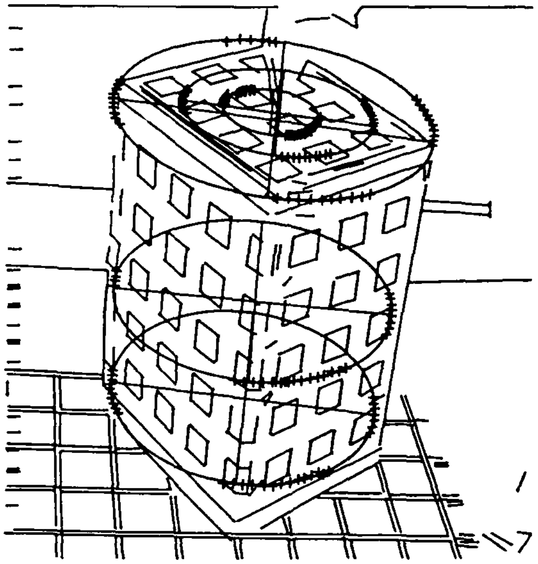


Fig. 13: Conic fits to linked 10-frame point tracks

Table 2: **Results for Eight sample points for the Box sequence**

Data Type		Axis dir.			Axis locn.			True Structure		Computed Structure	
		b_x	b_y	b_z	c_x	c_y	c_z				
True		0.102	-0.845	0.525	-0.051	0.522	0.851	d	k	d	k
Pt 1	1	-0.108	-0.839	0.533	0.057	0.530	0.846	0.458	0.142	0.483	0.139
	2	0.112	0.933	0.342	-0.050	-0.339	0.940				
Pt 2	1	0.110	-0.816	0.567	-0.061	0.564	0.824	0.575	0.116	0.638	0.117
	2	-0.101	0.851	0.515	0.075	-0.510	0.857				
Pt 3	1	-0.119	0.860	-0.497	0.054	-0.494	-0.868	0.681	0.113	0.634	0.109
	2	0.099	-0.809	-0.579	-0.086	0.573	-0.815				
Pt 4	1	0.110	-0.836	0.538	-0.055	0.535	0.843	0.459	0.094	0.475	0.093
	2	-0.112	0.941	0.319	0.051	-0.315	0.948				
Pt 5	1	0.137	-0.846	0.514	-0.061	0.511	0.857	0.458	0.039	0.446	0.040
	2	-0.137	0.946	0.295	0.061	-0.289	0.955				
Pt 6	1	0.083	-0.854	0.513	-0.036	0.512	0.858	0.628	0.128	0.608	0.128
	2	-0.069	0.842	0.535	0.058	-0.532	0.845				
Pt 7	1	0.103	-0.848	0.520	-0.052	0.517	0.854	0.458	0.083	0.449	0.081
	2	-0.107	0.949	0.297	0.043	-0.295	0.955				
Pt 8	1	0.134	-0.843	0.522	-0.072	0.517	0.853	0.733	0.129	0.722	0.127
	2	-0.109	0.749	0.654	0.105	-0.645	0.757				

On the Mathematical Foundations of Smoothness Constraints for the Determination of Optical Flow and for Surface Reconstruction *

M. A. Snyder
Computer and Information Science
University of Massachusetts
Amherst, Mass. 01003

March 29, 1989

Abstract

Gradient-based approaches to the computation of optical flow often use a minimization technique incorporating a smoothness constraint on the optical flow field. In this paper, we derive the most general form of such a smoothness constraint which is quadratic in first derivatives of the flow field, and quadratic in first or second derivatives of the grey-level image intensity function, based on three simple assumptions about the smoothness constraint: (1) that it be expressed in a form which is independent of the choice of Cartesian coordinate system in the image; (2) that it be positive definite; and (3) that it not couple different components of the optical flow. We show that there are essentially only four such constraints; any smoothness constraint satisfying (1,2,3) must be a linear combination of these four, possibly multiplied by certain quantities invariant under a change in the Cartesian coordinate system. Beginning with the three assumptions mentioned above, we mathematically demonstrate that all the best-known *smoothness constraints appearing in the literature are special cases* of this general form, and, in particular, that the "weight matrix" introduced by Nagel is essentially (modulo invariant quantities) the only physically plausible such constraint. We also show that the results of Brady and Horn on "rotationally symmetric" performance measures for surface reconstruction are simple corollaries of our main results, and in fact that such performance measures are invariant under the larger group of transformations consisting of rigid motions of the plane.

1 Introduction

The computation of optical flow from a pair of frames in a dynamic image sequence is an important problem in computer vision. One of the major techniques that has been developed to address this problem is to minimize the sum of two functionals, one based on the (local) intensity constancy constraint, and the other on a more global feature of the optical flow field, usually called a smoothness constraint.

It is generally known that the local intensity constraint does not uniquely determine the optical flow vector **U** at a point in the image. Along linear structures in the image only the component normal to the local edge direction may be determined, and at points in homogeneous areas even that information may not be available [Anan87].

The simplest example of this gradient-based approach is the work of Horn and Schunck [Horn81]. Here, the temporal variation $\partial I/\partial t$ of the grey-level image intensity function $I(x,y,t)$ at a fixed point in the image, and the spatial variation $\vec{\nabla} I$ of I at a fixed time are measured. These two quantities are related (under various assumptions-see [Horn81,Schu84a,Schu84b,Horn87]) to the optical flow $\vec{U}(x,y,t)$ via an image intensity constancy constraint:

$$\frac{\partial I}{\partial t} + \vec{U}\cdot\vec{\nabla} I = 0, \tag{1}$$

or in matrix notation,

$$I_t + \mathbf{U}^T \nabla I = 0. \tag{2}$$

*supported by DARPA/Army ETL under grant DACA76-85-C-0008, and by NSF under grant NSF/CER DCR8500332.

Here we have defined the matrix $\mathbf{U}^T = (U_x, U_y) \equiv (u, v)$, and $\nabla^T I \equiv (I_x, I_y)$; we denote the fact that some object is a matrix by using the corresponding bold face symbol. We represent the derivative of I with respect to the quantity ξ by $I_\xi = \partial I / \partial \xi$.

The single equation (2) is not, of course, sufficient to determine the quantity $\mathbf{U}$, since $\mathbf{U}$ has two components. Hence, it is clear that some additional constraint must be used to determine the optical flow field. Such a constraint typically demands some sort of consistency between neighboring flow vectors, i.e., it involves derivatives of $\mathbf{U}$. Such constraints are usually called "smoothness constraints."

A significant problem for such an approach, however, is that there are situations, such as at motion boundaries, where neighboring flow vectors need not be consistent in this sense. Smoothness constraints will therefore be expected to encounter difficulties near such boundaries. Indeed, the physical motivation for the work of Nagel and Enkelmann on "oriented smoothness constraints" [Nage86] was to try to supress such a constraint in the direction perpendicular to such boundaries.

Because of the corrupting effects of noise, aliasing, and other artifacts of the measurement process (as well as for the cases where the intensity-constancy constraint (2) is not valid even in the ideal case), it is not ordinarily appropriate to express this smoothness constraint as an additional equation, since these corrupting influences will customarily prevent (2) from being satisfied exactly anyway. This suggests an approach which looks for a flow $\overline{\mathbf{U}}$ which minimizes some combination of the degree to which $\mathbf{U}$ fails to satisfy the intensity constancy constraint (2), and the variation of $\mathbf{U}$ from "smoothness." There is no physical reason why such a $\mathbf{U}$ should be the "correct" flow which gave rise to the measured spatiotemporal image gradients, but we would expect that except in pathological cases, $\overline{\mathbf{U}}$ should approximate the "correct" $\mathbf{U}$.

The approach that has usually been taken is to minimize, over the space of all possible optical flow fields $\mathbf{U}$, a functional $\Pi[\mathbf{U}]$ given by

$$\Pi[\mathbf{U}] = \int \Sigma_{\rm int}\, dx\, dy + \int \Sigma_{\rm sm}\, dx\, dy, \tag{3}$$

where the integrals are over the image. Here

$$\Sigma_{\rm int} = \left(I_t + \mathbf{U}^T \nabla I\right)^2, \tag{4}$$

and $\Sigma_{\rm sm}$ is some quantity related to "smoothness." Since upon integration the quantity $\Sigma_{\rm sm}$ yields a number related to the deviation of the flow field from "smoothness," we will call it the *smoothness density* or, for simplicity, simply the *density*. Brady and Horn [Brad83] discuss this approach in some detail, and cite psychophysical evidence that something very like this is performed by some parts of the human visual system.

We note that the reason for choosing $\Sigma_{\rm int}$ as a quadratic functional (rather than, say, a quartic) is for mathematical simplicity only—a quadratic functional is convex and hence unimodal. In other words, such functionals are manifestly positive definite (for real functions), a desirable feature of any minimization approach.

The choice for $\Sigma_{\rm sm}$ is less obvious, and has less physical justification, since "smooth" is a vague concept. But, as we have stated, smoothness densities usually involve derivatives of $\mathbf{U}$. The density should also be positive definite.[1] Clearly, it is simplest to choose $\Sigma_{\rm sm}$ to involve only first derivatives of $\mathbf{U}$ (although other choices could be—and have been—made). We will call such constraints *first degree smoothness constraints.*

We will confine ourselves to first degree constraints in this work, leaving the question of second or higher degree constraints (such as the second degree constraint considered by Anandan and Weiss [Anan85]) to the sequel to this work [Snyd89b]. Smoothness densities have usually been chosen on the basis of either simplicity, or of heuristic arguments. In this paper, we proceed in the opposite way by defining the smoothness density mathematically, and then deriving all possible such smoothness densities.

We therefore consider the most general form of such a quadratic first degree density:

$$\Sigma_{\rm sm} = \sum_{ijk\ell} f_{ij}^{k\ell}\, \partial_i U_k \partial_j U_\ell \tag{5}$$

where $i, j, k, \ell = 1, 2$, with $\partial_i = \partial / \partial x_i$, $x_1 = x$, $x_2 = y$, $\vec{U} = (U_1, U_2) = (u, v)$ is the optical flow vector, and $f_{ij}^{k\ell}$ does not depend on $\mathbf{U}$ or its derivatives.

[1] The requirement is actually that the quantity be bounded from below, so as to guarantee the existence of a minimum. But any such quantity can be made into a positive definite function by adding an appropriate constant. Since the constant does not depend on $\mathbf{U}$ or its derivatives, both quantities give the same Euler-Lagrange equations. Hence we may, without loss of generality, assume the functional to be positive definite.

In order to make life bearable, we also introduce the *Einstein Summation Convention*: if in any product an index is repeated, it is to be understood that the index is to be summed over from 1 to 2. We therefore rewrite (5) as

$$\Sigma_{sm} = f_{ij}^{k\ell} \partial_i U_k \partial_j U_\ell . \tag{6}$$

Since i, j, k, and ℓ are repeated indices, it is understood that in (6) they are to be summed over. In the event that repeated indices in a product are not to be summed over, we will denote that by the phrase "(no sum)" next to the expression.

We will see in the next section that all of the smoothness densities so far proposed (see, e.g., [Horn81,Nage86]) satisfy the following three conditions:

1. They are invariant under a change of the Cartesian coordinate system in the image plane.
2. They are positive definite
3. They do not mix different components of **U**, i.e., the components of **U** are decoupled in (6).

We discuss the significance of each of these conditions in turn:

1. The condition that the smoothness density be invariant under a change of the Cartesian coordinate system of the image plane is equivalent to stating that the value obtained for the integral of Σ_{sm} over the image plane is independent of the coordinate system chosen for its evaluation. This seems eminently reasonable: the image itself has no preferred Cartesian coordinate system, so why impose one on it? This is equivalent to the condition that the Euler-Lagrange equations which follow from using this smoothness density are covariant under a change in coordinate system (i.e., that they have the same form in all Cartesian coordinate systems). We show in Appendix A that our condition is equivalent to the requirement that the density Σ_{sm} transform as a scalar under the action of the semi-direct product group ISO(2) of rigid transformations of the plane—the Euclidean group of the plane (see App. A). The situation in this respect is like the requirement of Galilean or Lorentz invariance in physics. Requiring that fundamental objects (like the Lagrangian) be invariant under some group of transformations results in the covariance of the resultant equations of motion under that same group of transformations.
2. The requirement of positive definiteness for Σ_{sm} is necessary to guarantee that $\Pi[\mathbf{U}]$ has a minimum, as discussed previously.
3. The requirement that the components of **U** be decoupled in Σ_{sm} has no physical basis I am aware of. We consider the effect of such a coupling in [Snyd89b]. This requirement is equivalent to demanding that $\Sigma_{sm}[\mathbf{U}] = \Sigma_{sm}[u] + \Sigma_{sm}[v]$.

In the next section, we elevate the properties (1,2,3) to the status of requirements for any smoothness constraint, and discuss the implications of this for the possible smoothness densities.

We believe that the demand of invariance under a change in the Cartesian coordinate system should be made not only of quantities like the smoothness density, but for any functional. We state this here as the "Zeroth Law of Computer Vision":

The 0^{th} Law of Computer Vision

Any functional having as domain functions defined over the image plane must be invariant under ISO(2)

2 The Definition of a Smoothness Density

2.1 General expression for the smoothness density

We will consider only smoothness densities of the form (6). We will require Σ_{sm} to satisfy the following Requirements:

I. Σ_{sm} is invariant under ISO(2).

II. Σ_{sm} is positive definite.

III. The structure of Σ_{sm} is such that the different components (u, v) of the optical flow field **U** are decoupled in all Cartesian coordinate systems. That is, Σ_{sm} can be written as the sum of two integrands, one which depends only on u, and the other only on v.

We have seen that requirement **I** is necessary in order that the integral have a unique value, independent of the coordinate system, for a given optical flow and image intensity. Requirement **II** is necessary in order to ensure that the smoothness integral have a minimum. Requirement **III** is not in any sense necessary, but it is characteristic of all the smoothness densities so far proposed; it is equivalent to assuming that the two components of the optical flow are "smoothed" independently.

Requirement **I** has as an immediate consequence that the integrand $\Sigma_{sm} \equiv f$ must be an ISO(2) scalar, i.e., that $f'(\mathbf{r}') = f(\mathbf{r})$, where $\mathbf{r}' = \mathbf{Rr} + \mathbf{t}$; here $\mathbf{R} = \mathbf{R}(\theta) \in SO(2)$ is the 2×2 rotation matrix, and $\mathbf{t}$ is the translational vector (see Appendix A of [Snyd89a]).

We show in [Snyd89a] that Requirements **I** and **III** together imply that the smoothness density must be of the form

$$f = \mathrm{tr}\left(\mathbf{\Omega}^T \mathbf{F} \mathbf{\Omega}\right), \tag{7}$$

where $\mathbf{F}$ transforms like a (second-rank) tensor under ISO(2), i.e.,

$$\mathbf{F} \longrightarrow \mathbf{F}' = \mathbf{RFR}^T. \tag{8}$$

Here, we have defined the tensor quantity $\mathbf{\Omega} = \nabla \mathbf{U}^T$. We will call the matrix $\mathbf{F}$ the *interaction* for the smoothness density f.

In [Snyd89a], we prove (Corollaries C.2.1 and C.3.1 of Appendix C of [Snyd89a]) that there is only one independent scalar-based interaction quadratic in 1st derivatives of the image intensity I, and only two independent such interactions quadratic in 2nd derivatives of I, as was originally shown by Brady and Horn [Brad83]. (Note, however, that they showed this only for the rotational subgroup SO(2) of ISO(2).)

3 The Determination of All Possible Smoothness Densities Quadratic in First or Second Derivatives of I

Since we have assumed that all the dependence of Σ_{sm} on derivatives of the flow field is contained in the matrix $\mathbf{\Omega}$, it follows that in the absence of any significant pre-processing such as grouping or model recognition, the interaction $\mathbf{F}$ can only depend on the grey-level image intensity function $I(x, y)$. If we are to limit ourselves to tensor-based interactions, then we must construct out of I and its derivatives objects which transform like ISO(2) tensors. This is done in the next two Sections for the case of 1st and 2nd derivatives of I. The reader should consult [Snyd89a] for all details of the derivations.

3.1 Interactions that are Quadratic in First Derivatives of I

Since I is a scalar, it follows that there are precisely two vectors which can be constructed from the (two) first derivatives of I, namely, ∇I and its dual vector $\mathbf{J}\nabla I = \tilde{\nabla} I$. It is shown in Theorem C.1 of Appendix C in [Snyd89a] that these are the only two independent vectors which can be so constructed. Now a second rank tensor must, in particular, have two indices (the row and column of the matrix which represents it). This means that the minimal tensor interaction must be quadratic in first derivatives of I. We know from Appendix A of [Snyd89a] that if $\mathbf{A}$ and $\mathbf{B}$ are vectors, then the outer product $\mathbf{AB}^T$ transforms as a tensor. Consequently, we can construct four tensors that are quadratic in 1st derivatives of I:

$$\nabla I \nabla^T I \equiv \mathbf{K} = \mathbf{K}^T = \begin{pmatrix} I_x^2 & I_x I_y \\ I_x I_y & I_y^2 \end{pmatrix} \tag{9}$$

$$\tilde{\nabla} I \nabla^T I = \mathbf{JK} = \begin{pmatrix} I_x I_y & I_y^2 \\ -I_x^2 & -I_x I_y \end{pmatrix} \tag{10}$$

$$\nabla I \tilde{\nabla}^T I = \mathbf{KJ}^T = (\mathbf{JK})^T = \begin{pmatrix} I_x I_y & -I_x^2 \\ I_y^2 & -I_x I_y \end{pmatrix} \tag{11}$$

$$\tilde{\nabla} I \tilde{\nabla}^T I \equiv \tilde{\mathbf{K}} = \mathbf{JKJ}^T = \begin{pmatrix} I_y^2 & -I_x I_y \\ -I_x I_y & I_x^2 \end{pmatrix} \tag{12}$$

The tensor $\tilde{\mathbf{K}} = \mathbf{JKJ}^T$ is called the *dual* of the tensor $\mathbf{K}$. Since $\mathbf{J} = \mathbf{R}(\pi/2)$, we see that $\tilde{\mathbf{K}}$ is just the tensor $\mathbf{K}$, rotated by 90°. In [Snyd89a], we prove the following Theorem, and its Corollary:

Theorem C.2 *If* $\mathbf{H}$ *is a tensor quadratic in* 1^{st} *order derivatives of* I*, then*

$$\mathbf{H} = a_0\mathbf{K} + a_1\mathbf{J}\mathbf{K} + a_2\mathbf{K}\mathbf{J}^T + a_3\tilde{\mathbf{K}},$$

where the tensors $\mathbf{K}, \ldots$ *are defined in (9)—(12), and where the* a_i *are constants.*

Corollary C.2.1 *If* H *is an ISO(2) scalar which is quadratic in* 1^{st} *derivatives of the image intensity* I*, then* H *is a multiple of* $I_x^2 + I_y^2$.

Therefore, the quantities (9)—(12) constitute a complete list of all the tensor-based interactions consistent with Requirements **I** and **III**.

We now impose the ancillary requirement (discussed in Section 2) that the interaction $\mathbf{F}$ be symmetric. It can be shown that only $\mathbf{K}$ and $\tilde{\mathbf{K}}$ lead to positive definite densities.

We summarize these results in the following theorem:

Theorem 1 *If a smoothness density satisfies Requirements* **I**, **II**, *and* **III**, *and is quadratic in* 1^{st} *derivatives of* I*, then the tensor-based interaction which gives rise to it must be of the form:*

$$\mathbf{F} = a_1\mathbf{K} + a_2\tilde{\mathbf{K}},$$

where a_1 *and* a_2 *are constants.*

3.2 Interactions Quadratic in Second Derivatives of I

In this section, we consider interactions which are quadratic in second derivatives of I, i.e., $\mathbf{F}$ is of the form

$$\mathbf{F} = \mathbf{A}^{(ijkl)}\partial_i\partial_j I\partial_k\partial_l I, \tag{13}$$

where $\mathbf{A}^{(ijkl)}$ are *matrices* (*not* matrix elements!).

We can construct such tensors by noting that the quantities

$$\nabla\nabla^T I \equiv \mathbf{L} = \begin{pmatrix} I_{xx} & I_{xy} \\ I_{xy} & I_{yy} \end{pmatrix}, \tag{14}$$

$$\tilde{\nabla}\nabla^T I = \mathbf{J}\mathbf{L} = \begin{pmatrix} I_{xy} & I_{yy} \\ -Ixx & -I_{xy} \end{pmatrix}, \tag{15}$$

$$\nabla\tilde{\nabla}^T I = \mathbf{L}\mathbf{J}^T = \begin{pmatrix} I_{xy} & -I_{xx} \\ I_{yy} & -I_{xy} \end{pmatrix}, \tag{16}$$

$$\tilde{\nabla}\tilde{\nabla}^T I = \mathbf{J}\mathbf{L}\mathbf{J}^T \equiv \tilde{\mathbf{L}} = \begin{pmatrix} I_{yy} & -I_{xy} \\ -I_{xy} & I_{xx} \end{pmatrix} \tag{17}$$

are all tensors which are linear in 2^{nd} derivatives of I. Hence, the $4 \times 4 = 16$ products of each of these tensors with each other are a set of tensors which are quadratic in 2^{nd} order derivatives of I.

We have no *a priori* guarantee, however, that all second rank tensors quadratic in 2^{nd} order derivatives of I can be obtained in this way. That is, the same question of completeness arises here as it did in the previous section. It can be shown [Snyd89a] that this set of 16 tensors is a complete set.

Upon imposing the requirement that the corresponding density be positive definite (Requirement **II**), one can show that only two of the sixteen interactions which are quadratic in 2^{nd} derivatives of I lead to positive definite densities (see [Snyd89a]), namely $\mathbf{L}^2$ and $\tilde{\mathbf{L}}$. This is summarized in the following theorem:

Theorem 2 *If* $\Theta = \mathrm{tr}\left[\mathbf{\Omega}^T\mathbf{F}\mathbf{\Omega}\right]$, *where* $\mathbf{F}$ *is quadratic in* 2^{nd} *derivatives of* I*, is a tensor-based smoothness density satisfying Requirements* **I**, **II**, *and* **III**, *then*

$$\mathbf{F} = a_0\mathbf{L}^2 + a_1\tilde{\mathbf{L}}^2,$$

where a_0 *and* a_1 *are constants.*

4 Relation to the Work of Brady and Horn on Performance Measures for Surface Reconstruction

Our claim that a smoothness density (or any other object, such as the "performance index" considered by Brady and Horn [Brad83]) be ISO(2) invariant generalizes the requirement proposed by these authors that such objects be "rotationally symmetric." We note that the central mathematical results of Brady and Horn (Propositions 2 and 6 in [Brad83], namely that $I_x^2 + I_y^2$ is the only independent "rotationally symmetric" scalar quadratic in 1st derivatives of I, and that $(\nabla^2 I)^2$ (the "squared Lagrangian") and $I_{xx}^2 + 2I_{xy}^2 + I_{yy}^2$ (the "quadratic variation") are the only independent "rotationally symmetric" quantities quadratic in 2nd derivatives of I appear (with "rotationally symmetric" replaced by the more general "ISO(2) scalar") as immediate corollaries (C.2.1 and C.3.1) of our main theorems. In addition, the cumbersome "tensor product" notation used by them is seen to be unnecessary, and to have a more elegant expression in terms of the tensors and scalars we have introduced here.

5 Relation to the Work of Nagel and Enkelmann

In searching for a smoothness constraint that would be more effective than the isotropic constraint of Horn and Schunck at computing optical flow near motion and depth boundaries, Nagel [Nage83,Nage87], and Nagel and Enkelmann [Nage86], noted that a suppression of the constraint along image gradients and along one of the two principal directions of the image intensity surface, would accomplish that task to a certain degree. Such a smoothness constraint is called by them an "oriented smoothness constraint." What we have called the "interaction" is called by them a "weight matrix." The interaction introduced by them is essentially a linear combination of the tensor interactions $\tilde{\mathbf{K}}$ and $\tilde{\mathbf{L}}^2$ we introduced in Section 3. Indeed, the matrices $\mathbf{F}$ and $\mathbf{C}^{-1}$ given in [Nage86] are given in our notation by

$$\mathbf{F} = \tilde{\mathbf{K}} + b^2\tilde{\mathbf{L}}^2 \tag{18}$$

$$\mathbf{C}^{-1} = \frac{\mathbf{F}}{\det \mathbf{F}}. \tag{19}$$

They also considered other normalizations of the interaction $\mathbf{F}$, such as dividing $\mathbf{F}$ by $\operatorname{tr}\mathbf{F}$. It is clear that since the determinant and trace of a tensor are ISO(2) invariant, such normalizations are simply multiplication of a tensor-based ISO(2) density by an ISO(2) scalar. Similar comments obtain for the later smoothness constraint discussed in the recent paper by Nagel [Nage88]. We note also that the smoothness constraint used by Hildreth [Hild83] is shown by Nagel [Nage87] to be a special case of the smoothness constraint (19).

We show in [Snyd89a] that the matrices $\mathbf{K}$ (and $\tilde{\mathbf{K}}$) can be viewed as projection operators that project any vector onto its component parallel (and perpendicular) to the local image gradient. The resultant smoothness density will then "smooth" only the components of the optical flow perpendicular (parallel) to the image iso-intensity contours. Although there is no physical basis for *demanding* smoothness along one of these directions, there is at least some justification for *not* demanding smoothness of the flow field components perpendicular to the image isointensity contours. Since these contours often (but not necessarily) correspond to physically meaningful (motion, occlusion) boundaries, it is often the case that the optical flow field varies strongly—perhaps even being discontinuous—at such boundaries. Consequently, it would seem that the interaction $\mathbf{K}$ should most definitely *not* be used for a smoothness interaction. (This argument is due to Nagel [Nage86].) This leaves only the quantity $\tilde{\mathbf{K}}$ as a possible smoothness interaction quadratic in 1st derivatives of the image intensity.

The matrices $\mathbf{L}$ and $\tilde{\mathbf{L}}$ are not projection operators, but rather, can be written as the sum of two projection operators. If we denote by $\mathbf{P}_1$ ($\mathbf{P}_2$), the projection operator which projects any vector onto its component parallel to the direction of maximum (minimum) principal curvature, respectively, then we show in [Snyd89a] that

$$\begin{aligned} \mathbf{L} &= \lambda_1\mathbf{P}_1 + \lambda_2\mathbf{P}_2 \\ \tilde{\mathbf{L}} &= \lambda_2\mathbf{P}_1 + \lambda_1\mathbf{P}_2, \end{aligned}$$

which implies that

$$\begin{aligned} \mathbf{L}^2 &= \lambda_1^2\mathbf{P}_1 + \lambda_2^2\mathbf{P}_2 \\ \tilde{\mathbf{L}}^2 &= \lambda_2^2\mathbf{P}_1 + \lambda_1^2\mathbf{P}_2. \end{aligned}$$

Here λ_1 (λ_2) is proportional to the maximum (minimum) principal curvature, respectively. As a consequence, a smoothness interaction using $\mathbf{L}^2$ will smooth the component of the optical flow along the direction of maximum principal curvature more strongly than the component along the direction of minimum principal curvature. Using $\tilde{\mathbf{L}}^2$ will do just the opposite. As for the previous case, the former seems to be the wrong thing to do, since the direction along which the principal curvature is a maximum often corresponds to a direction in which smoothness of the optical flow should not be demanded. This leaves $\tilde{\mathbf{L}}^2$ as a physically possible smoothness interaction quadratic in 2^{nd} derivatives of I.

We have therefore shown that the approach of Nagel and Enkelmann is the only physically reasonable approach (modulo ISO(2) invariant quantities) that can be taken for "orientation dependent" smoothness interactions quadratic in 1^{st} or 2^{nd} derivatives of I.

6 Conclusions

We have shown in this work that by using three simple and reasonable assumptions about the characteristics of smoothness constraints, there are essentially only 4 independent smoothness constraints that are quadratic in 1^{st} derivatives of the optical flow field, and quadratic in either 1^{st} or 2^{nd} derivatives of the grey-level image intensity function. Only two of these four are physically plausible, and they correspond to those chosen by Nagel and Enkelmann. All other such smoothness constraints can be obtained as linear combinations of these 4, perhaps multiplied by ISO(2) scalar functions of the image intensity and its derivatives.

We also derived generalized versions of the results of Brady and Horn on the possible performance measures that can be used for surface reconstruction, and found them to be simple corollaries of our main results for optical flow.

In the continuation of this work [Snyd89b], we investigate the more complicated problem of classifying smoothness densities quadratic in 2^{nd} derivatives of the optical flow field, and the implications of relaxing Requirement **III**, that the optical flow components are decoupled. Perhaps a coupling of these components in the smoothness constraint can lead to interesting smoothness constraints. For instance, the physically sensible smoothness constraint should reflect a smoothness in the three-dimensional flow field. Upon central projection, this will become a smoothness constraint on the two-dimensional optical flow. Because of the projection, such a two-dimensional smoothness constraint should be of the coupled variety. Consequently, perhaps it is the coupled smoothness constraints which are the most interesting [P. Anandan, personal communication]. We are presently investigating this idea.

Acknowledgements

I thank Prof. E. Riseman for his encouragement and support, and the members of the motion group at the University of Massachusetts for comments that led to the improvement of the ideas contained here. I also thank P. Anandan for several illuminating discussions.

References

[Anan87] P. Anandan, "A Unified Perspective on Computational Techniques for the Measurement of Visual Motion," Proc. 1^{st} Int. Conf. on Comp. Vis., London, England (June 1987).

[Anan85] P. Anandan and R. Weiss, "Introducing a Smoothness Constraint in a Matching Approach for the Computation of Displacement Fields," COINS Tech. Rep. 85-38, UMass, Amherst (Dec. 1985).

[Brad83] M. Brady and B. K. P. Horn, "Rotationally Symmetric Operators for Surface Interpolation," CVGIP **22**, 70-94 (1983).

[doCa76] M. P. do Carmo, *Differential Geometry of Curves and Surfaces*, Prentice-Hall (1976).

[Hild83] E. Hildreth, *The Measurement of Visual Motion*, MIT Press, Cambridge, Mass. (1983).

[Grim81] W. E. L. Grimson, *From Images to Surfaces: A Computational Study of the Human Early Visual System*, MIT Press, Cambridge, Mass. (1981).

[Horn87] B. K. P. Horn and E. J. Weldon, Jr. "Robust Direct Methods for Recovering Motion ," Univ. Hawaii at Manoa Tech. Rep. (Apr. 1987).

[Horn81] B. K. P. Horn and B. G. Schunck, "Determining Optical Flow," Art. Intell. **23**, 185-203 (1981).

[Nage83] H.-H. Nagel, "Constraints for the Estimation of Displacement Vector Fields from Image Sequences," Proc. IJCAI83, 945-951, Karlsruhe, W. Germany (1983).

[Nage86] H.-H. Nagel and W. Enkelmann, "An Investigation of Smoothness Constraints for the Estimation of Displacement Vector Fields from Image Sequences," IEEE Trans. Patt. An. Mach. Intell. **PAMI-8**, No. 5, 565-593 (Sept. 1986).

[Nage87] H.-H. Nagel, "On the Estimation of Optical Flow: Relations between Different Approaches and Some New Results," A:ι. Intell. **33**, 299–324 (1987).

[Nage88] H.-H. Nagel, "Image Sequences–Ten (Octal) Years–from Phenomenology towards a Theoretical Foundation," Int. Jour. Patt. Rec. and Art. Intell. **2** (3), 459–483 (1988).

[Schu84a] B. G. Schunck, "The Motion Constraint Equation and Surface Structure," Nat. Conf. Art. Intell., Austin, Tex. (1984).

[Schu84b] B. G. Schunck, "The Motion Constraint Equation for Optical Flow," Int. Conf. Patt. Rec., Montreal, Canada (1984).

[Snyd89a] M. A. Snyder, "On the Mathematical Foundations of Smoothness Constraints for the Determination of Optical Flow and for Surface Reconstruction," COINS Tech. Rep. 89-05, UMass., Amherst (1989).

[Snyd89b] M. A. Snyder, "The Mathematical Foundations of Smoothness Constraints for the Determination of Optical Flow and for Surface Reconstruction II: The Case of Coupled Components," COINS Tech. Rep., UMass, Amherst, in preparation.

[Stra86] G. Strang, *Introduction to Applied Mathematics*, Wellesley-Cambridge Press, Wellesley, Mass. (1986)

[Thor79] J. A. Thorpe, *Elementary Topics in Differential Geometry*, Springer Verlag, New York (1979).

The derivation of 3-D surface shape from shadows.

MICHAEL HATZITHEODOROU

Department of Computer Science
Columbia University
New York, NY 10027

Abstract. We study theoretical and implementation issues that arise when solving the shape from shadows problem. In this problem, the shadows created by a light falling on a surface are used to recover the surface itself. The problem is formulated and solved in a Hilbert space setting. We construct the spline algorithm that interpolates the data and show that it is the best possible approximation to the original function. The optimal error algorithm is implemented and a series of tests is shown. We additionally show that the problem can be decomposed into subproblems and each one can be solved independently from the others. This decomposition is suited to parallel computation and can result in considerable reductions in the cost of the solution.

1. Introduction.

Research in the various *Shape from X* areas has produced a series of methods that recover a surface from different types of information available about it. The information that is most widely used in these Shape from X methods includes depth values, shading, texture, etc.

We propose another approach for surface reconstruction. This is the *Shape from Shadows* problem. Assume that we have a surface that is lighted by a light source. The light source will cast shadows on this surface (see fig. 1.) Then the light will move to a new position where it will cast new shadows. We collect the different images of the shadowed surfaces, at these various times. From those we obtain the location of the start and the end of the shadow, plus some additional information about the surface function. Given any series of images containing shadows, we want to construct an algorithm that produces an approximation to the surface with the smallest possible error.

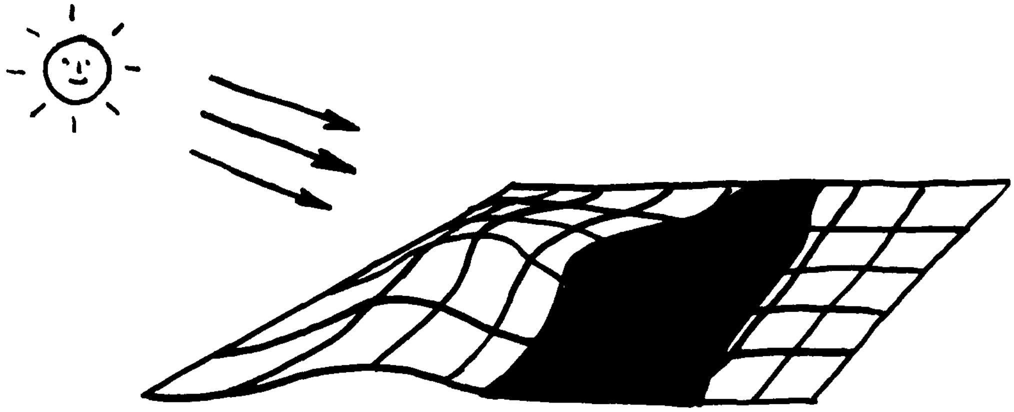


Figure 1.

Very little work has been done on the use of shadows for the reconstruction of the surface shape. In this paper we will extend the work presented in [5] where the solution to the 2-D version of the problem is obtained. In this 2-D version *surface slices*, intersections of the surface with planes perpendicular to the $x - y$ plane, were recovered. We generalize here the approach taken in [5] and present a method that will approximate the entire surface function, instead of a finite number of functions of one variable. The only other work we are aware of, is the one proposed in [6] where the problem is solved using a relaxation method.

Shadows are a very strong piece of information. The process that uses shadows is not affected by texture or by surface reflectance. Furthermore, our imaging system does not need a grey scale or color capabilities; it is sufficient for it to be able to distinguish between black and white. Also, noise in the form of bright spots inside a dark area and vice-versa can be filtered out easily. From the above, it is evident that shadows yield a powerful tool to be used in the reconstruction process.

Typeset by $\mathcal{A}\mathcal{M}\mathcal{S}$-TEX

We propose an algorithm that recovers the surface and minimizes the worst possible error within a factor of 2. The algorithm that minimizes the error is the spline algorithm.[1] We choose to construct the spline algorithm in steps using a process that converges to the optimal error algorithm. The justification behind this stepwise construction is that, in general, the optimal error algorithm might need many iterations to construct, but in most practical cases the initial version of the algorithm, or the one resulting from a few iterations, already obtains the optimal error. We therefore construct a process that has low cost, while achieving the smallest possible error.

We implement the optimal error algorithm and present a series of numerical runs so that we can see the performance of the algorithm in practice. The resulting approximations were very close to the function from which we obtained the data, and that can be immediately seen from the pairs of original and reconstructed surface that we provide. We also propose a parallel implementation of the algorithm that will considerably improve the running time.

The organization of the rest of the paper will be the following : In section 2, we will formulate the problem; we will define a function space in which our surface must belong. We will also define more precisely the information that can be extracted from the shadows.

In section 3, we will define the optimal error algorithm. We show that this is the spline algorithm which always exists and is unique. This will guarantee that the shape from shadows problem under this formulation is well-posed. The definition of a well-posed problem can be found in [3, 11]. In contrast to many other *shape from X* algorithms, our formulation does not require any regularization (see [9, 10] for a review of vision problems requiring regularization.) Understandably, if we suspect that the data are noisy, we might want to use a smoothing algorithm. In this case, the formulation that is produced by our approach is similar to the one that could be obtained from the application of regularization theory.

Section 4 deals with the implementation of the optimal algorithm. Its performance is analyzed in terms of the error it creates in recovering a surface. We will show sample runs that achieve a good approximation with a small number of data.

In section 5 we discuss the cost of the proposed algorithm. We show how to take advantage of the structure of the data and modify the algorithm, in order to obtain significant cost improvements. Furthermore, the algorithm can now be implemented in parallel, resulting in an even further reduction in the running time.

2. Formulation of the problem.

In the introduction of the paper we said that our aim is to recover a surface. Formally, a surface can be seen as a real function of two variables $f : \mathbb{R}^2 \longrightarrow \mathbb{R}$ belonging in the space of functions F_0. Our aim is to obtain an approximation $x \in F_0$ to our function $f \in F_0$ using the data that we can derive from the shadows. We want the approximation x to be as close to f as possible.

2.1 Function Space.

Let,

$$F_0 = \{f \mid f : [0,1]^2 \longrightarrow \mathbb{R},\ D^{1,1}f \text{ absolutely cont.}, \|D^{2,2}f\|_{L_2} \leq 1\}, \tag{2-1}$$

be the space that contains the functions f that we want to approximate.[2] [3] The norm $\|\cdot\|_{L_2}$ is defined as $\|f\|_{L_2} = \sqrt{\int_0^1 \int_0^1 |f(x,y)|^2 dxdy}$, and $D^{i,j} = \frac{\partial^{i+j}}{\partial x^i \partial y^j}$.

Also, define the bilinear form $\langle \cdot, \cdot \rangle$ to be such that,

$$\langle f, g \rangle = \int_0^1 \int_0^1 D^{2,2}f(x,y)\, D^{2,2}g(x,y)\, dxdy, \tag{2-2}$$

and the norm $\|\cdot\|$ to be such that,

$$\|f\| = \langle f, f \rangle^{1/2}. \tag{2-3}$$

Clearly $\langle \cdot, \cdot \rangle$ defined above is a semi-inner product and $\|\cdot\|$ is a semi-norm. If we pose the additional requirements $f(0,y) = 0$, $f(x,0) = 0$ and $D^{1,0}f(0,y) = 0$, $D^{0,1}f(x,0) = 0$ on our function, then the bilinear form $\langle \cdot, \cdot \rangle$ is an inner product and $\|\cdot\|$ a norm. Consequently, F_0 equipped with $\langle \cdot, \cdot \rangle$ is a semi-Hilbert space or a Hilbert space respectively.

2.2 Information.

In the next step we will extract from the image(s) the information, that is contained in the shadows, and which will be denoted by $N(f)$.[4] Assume that the light falls on the surface along the x-axis. Clearly, from the position

[1] We will define the *worst case error*, the *spline algorithm*, and all the other needed concepts later.

[2] The bound of 1 in $\|D^{2,2}f\|_{L_2}$ is assumed without loss of generality. However, as we already mentioned, any fixed bound is equally good.

[3] The use of the interval $[0,1]^2$ is not restrictive either. Any interval $[a,b] \times [c,d]$ for some a, b, c, and d is equally good.

[4] The concept of information is considerably different from the concept of data. The data vector is a vector of fixed values, while information is an operator. We will use the term somewhat imprecisely. The user is referred to [12, 13, 14, 15] for a more detailed discussion on the concept of information operators.

of the light source, we can immediately obtain the partial derivative of the function f, with respect to x, at the point (x_i, y_i), (x_i, y_i) being the beginning of the shadow (see fig. 2). We can also obtain the difference between the two function values $f(x_i, y_i) - f(x_i, y_i)$, at the beginning and at the end of the shadow respectively, given by $\frac{\partial f}{\partial x}(x_i, y_i)(x_i - x_i)$.

For a light falling on the surface along the y-axis we can obtain similar information. In particular, for a given shadowed area starting at (x_i, y_i), and ending at $(x_i, \bar{y}_i)$, we can obtain the partiai derivative of f with respect to y and the difference $f(x_i, y_i) - f(x_i, y_i)$ which is given by $\frac{\partial f}{\partial y}(x_i, y_i)(y_i - \bar{y}_i)$.

Assuming that $l_i(x, y)$ is the straight line segment passing through the points (x_i, y_i), (x_i, y_i), an additional piece of information can be obtained. It holds (see fig. 2) that,

$$f(x, y_i) \; < \; l_i(x, y_i), \quad \forall x \in [x_i, \bar{x}_i]. \tag{2-4}$$

If the light falls in the direction along the y-axis, then the obtained inequality is

$$f(x_i, y) \; < \; l_i(x_i, y), \quad \forall y \in [y_i, \bar{y}_i]. \tag{2-5}$$

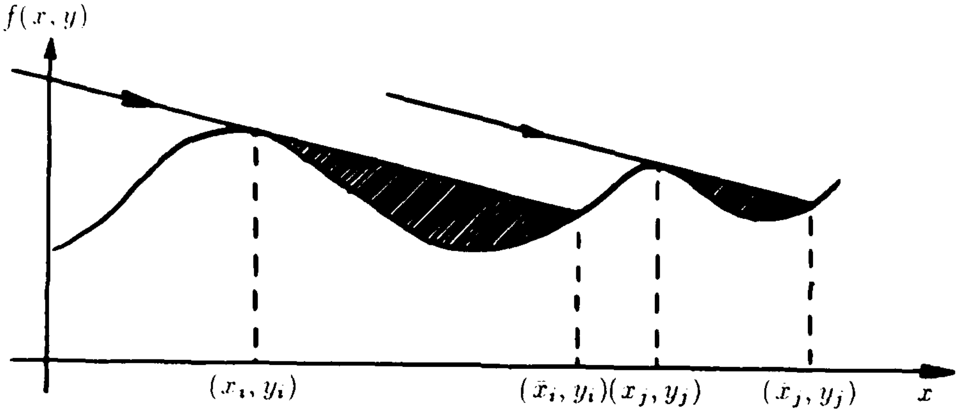


Figure 2.

So, formally, the information $N(f)$ contains triplets of the form,

$$\langle \frac{\partial f}{\partial x}(x_i, y_i), f(x_i, y_i) - f(\bar{x}_i, \bar{y}_i), f(x, y_i) < l_i(x, y_i) \rangle, \quad \bar{y}_i = y_i \tag{2-6}$$

or of the form,

$$\langle \frac{\partial f}{\partial y}(x_i, y_i), f(x_i, y_i) - f(\bar{x}_i, \bar{y}_i), f(x_i, y) < l_i(x_i, y) \rangle, \quad \bar{x}_i = x_i. \tag{2-7}$$

Note that the third item in the above triplets is a consistency condition.

In each one of the images in our sample there are 0, 1 or more shadowed areas. From each one of those shadowed areas we can obtain a triplet of the form (2-6) or (2-7). If we group all the data resulting from this sampling we obtain the vector,

$$\begin{aligned} N(f) = [\frac{\partial f}{\partial x}(x_1, y_1), \ldots, \frac{\partial f}{\partial x}(x_k, y_k), \frac{\partial f}{\partial y}(x_{k+1}, y_{k+1}), \ldots, \frac{\partial f}{\partial y}(x_n, y_n), \\ f(x_1, y_1) - f(x_1, y_1), \ldots, f(x_n, y_n) - f(\bar{x}_n, y_n), \\ f(x_1, y_1) - f(t_1, s_1), \ldots, f(x_1, y_1) - f(t_{m_1}, s_{m_1}), \ldots, \\ f(x_2, y_2) - f(t_{m_2}, s_{m_2}), \ldots, f(x_n, y_n) - f(t_{m_n}, s_{m_n})]^{T}, \end{aligned} \tag{2-8}$$

where $m_1, \ldots, m_n$ are the number of points (t_i, y) in every interval $[x_i, x_i] \times y$ for which (2-4) holds, or points (x, s_i) in $x \times [y_i, y_i]$ for which (2-5) holds, and $m_1 + \cdots + m_n = m$. Clearly, whi.e we know that there are exactly $2n$ pieces of information in the first part of $N(f)$, we cannot bound the cardinality of the last part because it can be the case that $m \longrightarrow +\infty$.

3. Solution of the problem - The optimal algorithm.

We now proceed to the solution of the problem. We want, given information $N(f)$, to obtain an algorithm that will provide an approximation to our function f. An *algorithm* φ is defined as any mapping from the space of all permissible data vectors to the space F_0.

3.1 Algorithm error.

The error of an algorithm for any fixed function f is given by $\|f - \varphi(N(f))\|$. We would like to know what is the largest possible error that can be made by the algorithm, i.e. we want the error of the algorithm for the worst possible function f.

DEFINITION 3.1. *The worst case error of an algorithm φ is,*

$$e(\varphi, N(f)) = \sup_{\bar{f} \in F_0} \left\{ \|\bar{f} - \varphi(N(\bar{f}))\|, \;\; N(\bar{f}) = N(f) \right\}. \tag{3-1}$$

Intuitively, the error of an algorithm is obtained by an adversary type of argument where the adversary chooses the function f so that the distance between f and φ is maximized.

Clearly, when solving any problem we would like to have an algorithm that will minimize $e(\varphi, N(f))$.

DEFINITION 3.2. *An algorithm φ^* that has the property,*

$$e(\varphi^*, N(f)) = \inf_{\varphi} \left\{ e(\varphi, N(f)) \right\}, \qquad \forall f \in F_0 \tag{3-2}$$

is called a strongly optimal error algorithm.

The quantity at the right side of (3-2), i.e. the infimum of the error of all algorithms solving the problem given information $N(f)$, is a property of the problem itself, and does not depend on the particular algorithm used at any moment. This quantity, gives the inherent uncertainty of the problem for given information, and is called the *radius of information*. Clearly, the error of the strongly optimal algorithm equals the radius of information.[5]

3.2 The spline algorithm.

We propose the spline algorithm φ^s for the solution of our problem. Splines have been known to give the optimal solution to many interesting problems [1, 2, 7, 8, 12, 13].

DEFINITION 3.3. *A spline σ is an element in the space of functions F_0 such that,*

(1) $N(\sigma) = \vec{y}$.

(2) $\|\sigma\| = \min_{f \in F_0}\{\|f\|, \;\; N(f) = \vec{y}\}$.

The meaning of (1) is that the spline must interpolate the data, and (2) says that the spline is the function that minimizes $\|\cdot\|$. The spline algorithm is the process that constructs the spline.

In a Hilbert space setting one can obtain a closed form for the spline algorithm. In our particular case, the spline algorithm is given by,

$$\varphi^s(x, y) = \sum_{i=1}^{2n} a_i g_i(x, y) + \sum_{j=1}^{m} c_j h_j(x, y), \tag{3-3}$$

where $\{g_i\}_{i=1,\dots,2n}$ and $\{h_j\}_{j=1,\dots,m}$ are such that,

$$D^{2,2} g_i(x, y) = \frac{(x_i - x)_+^0 (y_i - y)_+ - (x_{i-1} - x)_+^0 (y_{i-1} - y)_+}{\sqrt{x_i - x_{i-1}}}, \qquad i = 1, \dots, k \tag{3-4}$$

when the light falls along the x-axis, and

$$D^{2,2} g_i(x, y) = \frac{(y_i - y)_+^0 (x_i - x)_+ - (y_{i-1} - y)_+^0 (x_{i-1} - x)_+}{\sqrt{y_i - y_{i-1}}}, \qquad i = k + 1, \dots, n \tag{3-5}$$

for light along the y-axis, where $(a - b)_+^0 = 1$ for $a = b$ and 0 otherwise,

$$D^{2,2} g_{n+i}(x, y) = (\tilde{x}_i - x)_+ (\bar{y}_i - y)_+ - (x_i - x)_+ (y_i - y)_+ - (x_i - x)_+^0 (\bar{x}_i - x_i)(y_i - y)_+, \qquad i = 1, \dots, k \tag{3-6}$$

[5] One point that must be mentioned here is that the radius of information describes, as we said before, the *inherent uncertainty of the problem* and has a specific value, say R. However small or large R may be, there is no procedure that will guarantee error less than that.

$$D^{2,2}g_{n+i}(x,y) = (\bar{y}_i - y)_+(\bar{x}_i - x)_+ - (y_i - y)_+(x_i - x)_+ - (y_i - y)^0_+(\bar{y}_i - y_i)(x_i - x)_+, \qquad i = k+1,\ldots,n \quad \text{(3-7)}$$

where $(a-b)_+ = a - b$ for $a > b$ and 0 otherwise, and

$$D^{2,2}h_j(x,y) = (t_j - x)_+(s_j - y)_+ - (x_i - x)_+(y_i - y)_+ - (x_i - x)^0_+(t_j - x_i)(y_i - y)_+, \qquad \text{some } i, \quad j = 1,\ldots,m, \quad \text{(3-8)}$$

$$D^{2,2}h_j(x,y) = (s_j - y)_+(t_j - x)_+ - (y_i - y)_+(x_i - x)_+ - (y_i - y)^0_+(s_j - y_i)(x_i - x)_+, \qquad \text{some } i, \quad j = 1,\ldots,m. \quad \text{(3-9)}$$

The functions $\{g_i\}_{i=1,\ldots,2n}$ and $\{h_j\}_{j=1,\ldots,m}$ are the *representers* of the functionals that construct the information $N(f)$, properly modified to have a small area of support.

The coefficients a_i and c_j are chosen so that the definition of the spline is satisfied, that is, φ^s interpolates the data, and also minimizes the norm $\|\cdot\|$. If the area of support of every g_i is disjoint from the area of support of every other and furthermore $\langle g_i, g_j\rangle = \delta_{ij}$, the Krönecker delta, then the coefficients a_i are given directly by the theory. This is not the case in this setting where the coefficients a_i are obtained by solving a system of linear equations and the coefficients c_j are obtained by directly minimizing $\|\cdot\|$. We will describe the implementation of the algorithm in section 4. The derivation of the minimization problem is somewhat tedius and can be found in the appendix.

For the spline algorithm the following very strong theorem holds [8, 13].

THEOREM 3.1. *Let F_0 be a Hilbert space, $f \in F_0$ and information $\bar{y} = N(f)$. Then, the spline algorithm interpolating the data $\bar{y}$ exists, is unique, and achieves error at most twice the radius of information.*

From Theorem 3.1 we can obtain two very important results. First, our problem under the proposed formulation is well-posed. This property [3, 11] is always desirable when solving a problem. Computer vision problems tend to be ill-posed and considerable effort has been spent by the vision community towards the correct formulation that will yield well-posedness (See [4, 9, 10] for a survey.)

Second, the spline algorithm has a worst case error that is within a factor of 2 from the radius of information. The algorithm that achieves that, is called *almost strongly optimal* [12]. If the problem is *linear*[6] then the spline algorithm φ^s has a worst case error equal to the radius of information and is, therefore, the strongly optimal algorithm.

We can choose to ignore the existence of the inequalities (2-4) and (2-5) and not include them in the information $N(f)$. If we choose to do so, we essentially assume that $m = 0$ and the shape from shadows problem is a linear problem. Then, the spline algorithm becomes

$$\varphi^s(x,y) = \sum_{i=1}^{2n} a_i g_i(x,y). \quad \text{(3-10)}$$

As expected, the part $\sum_{j=1}^{m} c_j h_j(x,y)$ can now be omitted.

If on the other hand, $m > 0$ then, the problem is non-linear, and φ^s is an almost strongly optimal algorithm.[7]

We saw before that the construction of the algorithm has to be done in steps. First, we obtain the coefficients a_i and subsequently the coefficients c_j. We also mentioned that the cardinality of the non-linear part of the information m is not known a-priori. We would like to keep its value low to reduce the costs involved in solving the minimization problem. To obtain this we propose to break down the implementation of the algorithm in additional steps.

4. Application of the algorithm - Numerical runs.

The spline algorithm of section 3 has been applied and its performance has been tested in practice.

4.1 Algorithm implementation.

From our early experience we have concluded that in many cases, the non-linear part of the information is not needed.

Stage 1:

[6] For an exact definition of a linear problem see [8, 13, 14, 15].

[7] *Up to this moment there does not exist a general theory that will help construct strongly optimal algorithms for non-linear problems.* These algorithms are in general difficult to construct and are derived on a per problem basis.

Therefore, we begin the implementation of the spline algorithm by assuming that $m = 0$. We first construct the values of the coefficients a_i. This is done by solving the system of equations,

$$\mathbf{G}\,\vec{a} = \vec{y}, \tag{4-1}$$

where $\mathbf{G} = \{\langle g_i, g_j\rangle\}_{i,j=1}^{2n}$ and $\{g_i\}_{i=1,\dots,2n}$ are given by (3-4), (3-5), (3-6) and (3-7). The system is solved by a direct method without the need for pivoting since it is symmetric, positive definite and has a nice structure that reduces the number of calculations.

As a next step, we use the computed values of the a_i' s to construct the spline algorithm.

Third, we check to see whether the non-linear constraints are violated.

If the non-linear constraints (2-4) and (2-5) are not violated, which is usually the case, we do not need to do anything else. We have already obtained the approximation $\varphi^s(x,y)$ to the function f.

Since we have not used the non-linear part of the information, the problem is linear hence the spline algorithm achieves the radius of information.

Stage 2:

If the constraints (2-4) or (2-5) are violated, then we do not have a sufficiently good approximation, which means that we must obtain the coefficients c_j of (3-3). To do so, we have to solve the minimization problem derived in section 3.3.

We will consequently proceed as follows. We will take a few points (t_i, s_i) in the shadowed intervals where the constraints are violated. For these points we solve the minimization problem. Then we check again for violations of the non-linear constraints. If there are violations we repeat Stage 2. We select a few more points from the new interval(s) in which (2-4) or (2-5) are violated, and we add them to the sample. The minimization is repeated for the new set of points and the new coefficients are derived. At the same time, the a_i' s and the old c_j' s are modified.

We perform the minimization for a few points at a time for various reasons.

(1) Theoretically, the cardinality of the non-linear information m can be arbitrarily large. In practice though, we rarely need to use more that one or two points per shadowed area. Taking a few points at a time we can minimize the cost of the algorithm.

(2) At each new iteration we do not need to undo our previous work, but we simply modify the existing coefficients while deriving the new ones.

4.2 Test runs.

We have constructed a broad series of functions, and have run the algorithm using these as test surfaces.

From early test runs, we have observed that smooth functions can be approximated easily with almost non-observable error, using a small number of sample points.

The functions that are the most difficult to approximate, are the ones that have as few derivatives as possible. In the class F_0 these functions are piecewise quadratic polynomials which are constructed as the product of quadratic polynomials of one variable. We will show the performance of the algorithm φ^s on these functions.

We start the series of test runs with a function consisting of 100 polynomial pieces. We use two different light angles from each direction, two along the x-axis and two along the y-axis. The function and the reconstruction can be seen in figure 3.

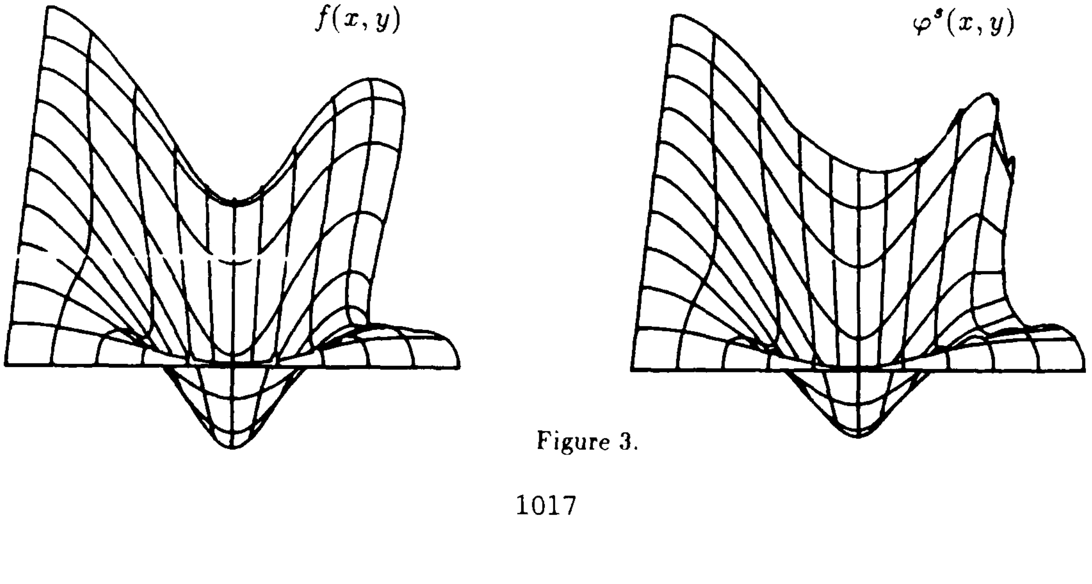


Figure 3.

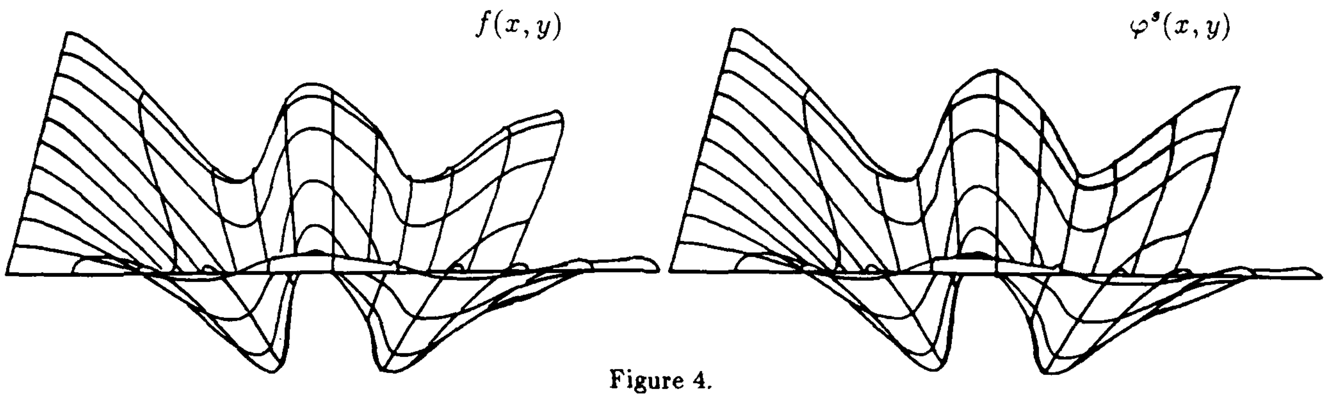


Figure 4.

In figure 4 we show a function consisting of 200 polynomial pieces. We again draw the reconstruction together with the function for comparison purposes. The information was obtained by using 4 different lighting angles in each direction.

Finally, in figure 5 we show one of the most difficult functions that we have constructed. It consists of 400 polynomial pieces with large jumps in the second derivatives.[8] The function has been approximated using a sample created from lights falling from six different light angles in each direction.[9]

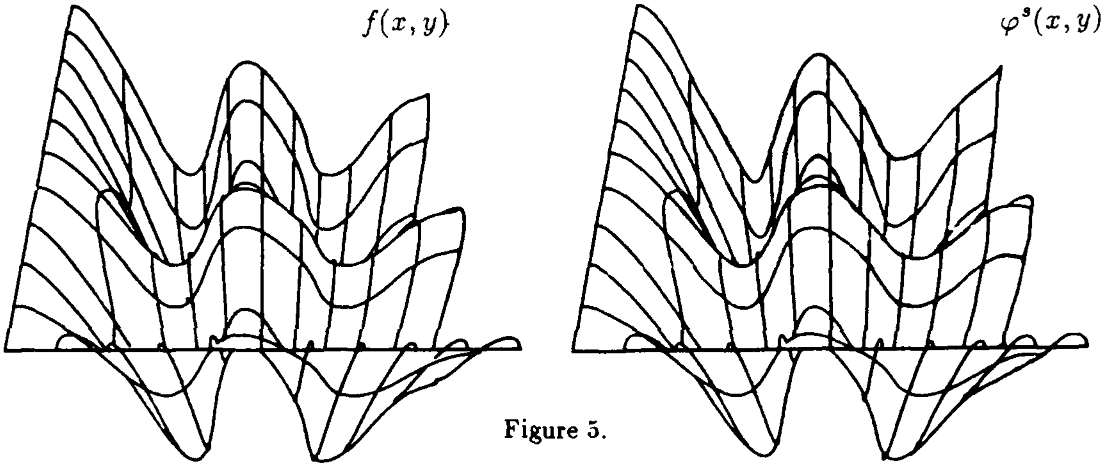


Figure 5.

5. Cost of the algorithm - Speed improvements.

5.1 Algorithm cost.

Let us now discuss the speed performance of our algorithm. The spline algorithm, as defined in section 3, is linear in terms of its input. Thus, if we knew the coefficients a_i and c_j then, $cost(\varphi^s)$ would be $O(n)$.

In our case, the coefficients of the spline algorithm are not known, and must be constructed. To achieve this we must solve a system of linear equations, and sometimes, a minimization problem. These costs dominate the cost of the algorithm.

In particular the solution of the system (4-1) has a cost $O(n^3)$. The cost of the non-linear minimization is considerably higher in the general case. Due to the special structure of the problem, whose derivation we show in the appendix, we can use a variation of the feasible directions approach. The proposed method has a cost of $O(n)$. We are in the process of investigating the properties of this method.

5.2 Speed improvements.

In section 5.1 we have discussed the cost of a very straightforward implementation of the spline algorithm described in section 4.1. We now show that a slight improvement in the implementation of the algorithm can yield a significant speedup. This speedup can be achieved only if the function we want to recover can be split in distinct sections that we will from now on call *valleys*. A valley is defined by two local maxima of the function, but also depends on the specific sampling. For example, the function of figure 4 has 2 valleys. In particular, we say that the function f, under some fixed sampling, has k valleys if we can define k partitions $\Pi_1, \Pi_2, \ldots, \Pi_k$ of the functions $\{g_i\}_{i=1,\ldots,2n}$,

[8] Which means that $\|D^{2,2}f\| \leq 1$ does not hold. Instead, $\|D^{2,2}f\| \leq C$, for large C holds. The mathematical formulation does not change, but the visual effect is noticeable.

[9] Theoretically, light falling from one direction only could be used, but in this case the problem is identical to the 2-D one [5].

given by (3-4) and (3-5), such that the union of the areas of support of all the functions in each partition is disjoint from the union of the areas of support of the functions in every other partition.

We can detect the existence of any number of valleys in time $O(n)$ and subsequently, we can solve k problems of sizes $n_1, n_2, \ldots, n_k$ respectively, instead of solving one problem of size n, where $n = n_1 + n_2 + \cdots + n_k$.

To connect the pieces resulting from each of the k problems we need constant time per problem, hence combining can be done in time $O(k)$.

Therefore, the total cost of this algorithm, which we will denote φ_i^s, will be $O(k\nu^3)$, where $\nu = \max\{n_1, \ldots, n_k\}$.

5.3 Parallel implementation.

Since splitting the problem into individual subproblems and combining the resulting surfaces is straightforward and cheap to implement, one immediate extension to the above set-up of the problem is to assign one individual subproblem to a different processor and solve the initial problem in a parallel or in a distributed environment.

Again, splitting into k subproblems requires time $O(n)$ and combining the individual solutions into one requires time of $O(k)$. Then every processor will require time $O(n_i^3)$, $i = 1, \ldots, k$ resulting in a total cost for the parallel version φ_p^s of our algorithm of $O(\nu^3)$, where $\nu = \max\{n_1, \ldots, n_k\}$.

6. Conclusion - Future work.

There are certain issues that we would like to investigate in the future. We would like to study *optimal* and *adaptive information*. The study of optimal information will determine the light placement that will minimize the error of the algorithm for a fixed number of samples. Having adaptive information will permit, given a number of samples, to choose where to place a new light so as to reduce the error as much as possible.

Although the work presented in this paper is not directly related to signal processing, the construction of a complete system will also have to address many issues that arise when the information is obtained from the shadows.

We solved the problem of recovering a surface from the shadows it casts on itself when lighted by a light source positioned at various locations.

We proposed a formulation that results in a well-posed problem and we have consequently proceeded into solving it. We proposed an optimal error algorithm which additionally achieves a low time cost, especially if a clever but simple breakdown of the problem is used.

The work described in this paper, extends the results on the reconstruction of 2-D surface slices. This new approach can be very desirable, especially if our purpose is to recover the entire shape of the surface.

The 2-D model can still be used if our aim is to recover the function value at just one point, in which case we only have to use the slice passing through this point. Also, if we only have lights along one axis then, the 2- and 3-D models become equivalent.

7. Acknowledgements.

I would like to thank Prof. John Kender for many valuable suggestions and discussions and Prof. Greg Wasilkowski, currently with the University of Kentucky at Lexington, for his guidance during the initial stages of this work.

I. Appendix - The minimization problem.

We want to minimize $||\sigma||^2$ where σ is given by (3-3). We can write,

$$\begin{aligned} ||\sigma||^2 &= \langle \sigma, \sigma \rangle \\ &= \sum_{i_1=1}^{n} \sum_{i_2=1}^{n} a_{i_1} a_{i_2} \langle g_{i_1}, g_{i_2} \rangle + 2 \sum_{i=1}^{n} \sum_{j=1}^{k} a_i c_j \langle g_i, h_j \rangle + \sum_{j_1=1}^{k} \sum_{j_2=1}^{k} c_{j_1} c_{j_2} \langle h_{j_1}, h_{j_2} \rangle \\ &= \vec{a}^{\mathsf{T}} \mathbf{G}\, \vec{a} + 2\, \vec{a}^{\mathsf{T}} \mathbf{P}^{\mathsf{T}} \vec{c} + \vec{c}^{\mathsf{T}} \mathbf{H}\, \vec{c}, \end{aligned} \tag{I-1}$$

where $\mathbf{G} = \{\langle g_i, g_j \rangle\}_{i,j=1,\ldots,2n}$, $\mathbf{P} = \{\langle h_j, g_i \rangle\}_{\substack{j=1,\ldots,m \\ i=1,\ldots,2n}}$, and $\mathbf{H} = \{\langle h_i, h_j \rangle\}_{i,j=1,\ldots,m}$.

Also,

$$\begin{aligned} \langle \sigma, g_s \rangle &= \sum_{i=1}^{n} a_i \langle g_i, g_s \rangle + \sum_{j=1}^{k} c_j \langle h_j, g_s \rangle = y_s \\ \Longrightarrow &\qquad \mathbf{G}\, \vec{a} + \mathbf{P}\, \vec{c} = \vec{y} \\ \Longrightarrow &\qquad \vec{a} = \mathbf{G}^{-1} (\vec{y} - \mathbf{P}\, \vec{c}), \end{aligned} \tag{I-2}$$

and,

$$\begin{aligned}
\langle \sigma, h_s\rangle = \sum_{i=1}^{n} a_i \langle g_i, h_s\rangle + \sum_{j=1}^{k} c_j \langle h_j, h_s\rangle \leq A_s& \\
\Longrightarrow \qquad \mathbf{P}^{\mathsf{T}}\vec{a} + \mathbf{H}\,\vec{c} \leq \vec{A}& \\
\overset{(\text{I-2})}{\Longrightarrow} \quad \mathbf{P}^{\mathsf{T}}\mathbf{G}^{-1}\vec{y} - \mathbf{P}^{\mathsf{T}}\mathbf{G}^{-1}\mathbf{P}\,\vec{c} + \mathbf{H}\,\vec{c} \leq \vec{A}& \\
\Longrightarrow \quad (\mathbf{H} - \mathbf{P}^{\mathsf{T}}\mathbf{G}^{-1}\mathbf{P})\,\vec{c} \leq \vec{A} - \mathbf{P}^{\mathsf{T}}\mathbf{G}^{-1}\vec{y}.&
\end{aligned} \qquad (\text{I-3})$$

Now, if we substitute (I-2) for $\vec{a}$ in (I-1) we obtain,

$$\begin{aligned}
\|\sigma\|^2 &= \vec{c}^{\mathsf{T}}\mathbf{H}\,\vec{c} + 2\left(\mathbf{G}^{-1}(\vec{y} - \mathbf{P}\vec{c})\right)^{\mathsf{T}}\mathbf{P}\,\vec{c} + \left(\mathbf{G}^{-1}(\vec{y} - \mathbf{P}\vec{c})\right)^{\mathsf{T}}\mathbf{G}\ \left(\mathbf{G}^{-1}(\vec{y} - \mathbf{P}\vec{c})\right) \\
&= \qquad \dots \\
&= \vec{c}^{\mathsf{T}}\mathbf{H}\,\vec{c} - \vec{c}^{\mathsf{T}}\mathbf{P}^{\mathsf{T}}\mathbf{G}^{-1}\mathbf{P}\,\vec{c} + \vec{y}^{\mathsf{T}}\mathbf{G}^{-1}\vec{y} \\
&= \vec{c}^{\mathsf{T}}\,(\mathbf{H} - \mathbf{P}^{\mathsf{T}}\mathbf{G}^{-1}\mathbf{P})\,\vec{c} + \vec{y}^{\mathsf{T}}\mathbf{G}^{-1}\vec{y}.
\end{aligned} \qquad (\text{I-4})$$

Since $\vec{y}^{\mathsf{T}}\mathbf{G}^{-1}\vec{y}$ has a known fixed value for any given problem, it remains to minimize $\vec{c}^{\mathsf{T}}\left(\mathbf{H} - \mathbf{P}^{\mathsf{T}}\mathbf{G}^{-1}\mathbf{P}\right)\vec{c}$ given the conditions in (I-3). This is a non-linear minimization problem. We can solve this problem using a feasible directions method. The matrix $\mathbf{Q} = \left(\mathbf{H} - \mathbf{P}^{\mathsf{T}}\mathbf{G}^{-1}\mathbf{P}\right)$ is positive definite, hence the problem is convex. Therefore, we are guaranteed to find the global minimum. At the same time, it can be noticed, that the constraints that we have are simple constant bounds on the variables which permits us to speed up the minimization method considerably. We are currently working on an even faster method for the solution of this problem.

References

[1] Anselone, P. M., and Laurent, P. J., *A general method for the const"uction of interpolating or smoothing spline functions*, Nurnner. Math. **12** (1968).

[2] Atteia, M., *Fonctions spline généralisées*, C.R. Acad. Sci. Paris **261** (1965).

[3] Hadamard, J., *Sur les problèmes aur derivées partielles et leur signification physique*, Princeton Univ. Bulletin **13** (1902).

[4] Hatzitheodorou, M. G., *The Application of Approximation theory methods to the solution of computer vision problems*, Columbia University, Comp. Science dep. (1988).

[5] Hatzitheodorou M. G., and Kender, J. R., *An optimal algorithm for the derivation of shape from shadows*, Proceedings IEEE CVPR (1988).

[6] Kender, J. R., and Smith, E. M., *Shape from darkness. Deriving surface information from dynamic shadows*, Proceedings AAAI (1986).

[7] Holmes, R., *R-splines in Banach spaces : I. Interpolation of linear manifolds*, J. Math. Anal. Appl. **40** (1972).

[8] Michelli, C. A., and Rivlin, T. J, *A Survey of optimal recovery*, in "Optimal estimation in Approximation theory," Plenum Press, 1977.

[9] Poggio, T., *Computer vision*, MIT Artificial Inteligence Lab (1986).

[10] Poggio, T., and Torre, V., *Ill-posed problems and regularization analysis in early vision*, MIT Artificial Inteligence Lab (1984).

[11] Tikhonov, A. N., and Arsenin, V. Y., "Solutions of ill-posed problems," V.H.Winston and Sons, 1977.

[12] Traub, J. F., Wasilkowski, G., and Woźniakowski, H., "Information, Uncertainty, Complexity," Addison-Wesley, 1983.

[13] Traub, J. F., and Woźniakowski, H., "A general theory of optimal algorithms," Academic Press, 1981.

[14] Woźniakowski, H., *A survey of information-based complexity*, Journal of Complexity **1** (1985).

[15] Woźniakowski, H., *Information-based complexity*, Annual Review of Computer Science **1** (1986).

STOCHASTIC STEREO MATCHING ON THE CONNECTION MACHINE

Stephen T. Barnard
Artificial Intelligence Center
SRI International
333 Ravenswood Avenue
Menlo Park, California 94025

Abstract

A stochastic approach to stereo matching is presented.[1] A microcanonical version of simulated annealing is used to approximate the ground states of a thermodynamic model system. The potential energy of the system combines two measures of the quality of a dense, two-dimensional disparity map: (1) the photometric error between corresponding points, and (2) the first-order variation (the "flatness") of the map. The method operates over a series of increasingly finer spatial scales. The implementation of this method on the Connection Machine^{tm} is discussed.

Introduction

Compared to other modes of depth perception, stereo vision seems relatively straightforward. The images received by two eyes are slightly different due to binocular parallax; that is, they exhibit a disparity that varies over the visual field, and that is inversely related to the distance of imaged points from the observer. If we can determine this disparity field we can measure depth and mimic human stereo vision. Few problems in computational vision have been investigated more intensively.

We describe an approach to stereo in which the matching problem is posed as computational analogy to a thermodynamic physical system. The state of the system encodes a disparity map that specifies the correspondence between the images. Each such state has an energy that provides a heuristic measure of the "quality" of the correspondence. To solve the stereo matching problem, one looks for the ground state, that is, the state (or states) of lowest energy. This paper is a condensation and revision of an earlier paper [1] and emphasizes some aspects of the implementation on the Connection Machine.^{tm}

Several features of the Connection Machine naturally fit computational vision problems of this type:

- The most important feature is its massive parallelism — up to 64K individual processors, each with 64K bits of memory. Many vision tasks are most naturally expressed as optimization problems on two-dimensional lattices of typically $256 \times 256 = 64K$ pixels.
- The Connection Machine architecture is flexible. It does not restrict the user to a fixed lattice size, or even to one- or two-dimensional lattices. In general, n-dimensional lattices are supported.
- Another attractive feature of the Connection Machine is its general method of interprocessor communication. Two varieties of message-passing networks are provided: a boolean n-cube router for general communication and a "NEWS" grid for communication between processors arranged in regular lattices.
- Finally, and not to be underestimated, is the excellent user interface of the Connection Machine, including language processors that are straightforward extensions of standard languages[2] and a graphic display system that provides a high-speed "window" into the system's memory.

[1] Support for this work was provided by the Defense Advanced Research Projects Agency under contracts DCA76-85-C-0004 and MDA903-83-C-0084.

[2] We use *Lisp, which is an extension of Common Lisp. Parallel versions of Fortran and C are also available.

The Stereo Matching Problem

Most approaches to stereo matching can be divided into three classes: correlation, feature-matching, and lattice models. Correlation, in its basic form, is the most obvious. Intensity patches in one image are matched to patches in the other image with search, typically using a normalized cross-correlation as a measure of similarity or a normalized mean-square-difference as a measure of dissimilarity. Many variations of this basic theme have been explored. The feature-matching approach matches directly between discrete sets of points — typically, the output of an edge detector, such as zero-crossing contours. Both suffer from similar difficulties:

- The size of the correlation patch or the support of the feature operator affects the likelihood of false matches. A correlation patch must be large enough to contain the information necessary to specify another patch unambiguously or, failing this, some additional means of disambiguating false matches must be used. Similarly, feature operators with small support will detect many features.
- At the same time, the correlation patch or the feature-operator support must be small compared to the variation in the disparity map. If either is too large, the system will be insensitive to significant relief in the scene.
- In typical images much of the area consists of uniform or slowly varying intensity where neither correlation nor feature matching will be effective.

Lattice models pose the stereo matching problem in terms of optimizing a measure that is usually interpreted as the energy of a lattice of interacting elements. To take one example, Julesz proposed a model consisting of two lattices of spring-loaded magnetic dipoles, representing the two images of a random-dot stereogram [2]. The polarity of the dipoles represents whether pixels in the left and right images are black or white. A state of global fusion is achieved in the ground state, with the attraction or repulsion of the dipoles balanced by the forces of the springs.

A Scaled-Lattice Model

The Stereo Energy Equation

Consider the following equation:

$$\mathcal{E} = \int\int \{(\mathcal{L}(x - \frac{\mathcal{D}}{2}, y) - \mathcal{R}(x + \frac{\mathcal{D}}{2}, y))^2 + \lambda(\nabla\mathcal{D})^2\} dx dy \ , \quad (1)$$

where $\mathcal{L}$ and $\mathcal{R}$ are piecewise-continuous intensity functions of the left and right visual fields, $\mathcal{D} = \mathcal{D}(x, y)$ is a cyclopean disparity map, and λ is a constant. Each value of $\mathcal{D}$ specifies two corresponding points: $(x - \mathcal{D}/2, y)$ and $(x + \mathcal{D}/2, y)$.

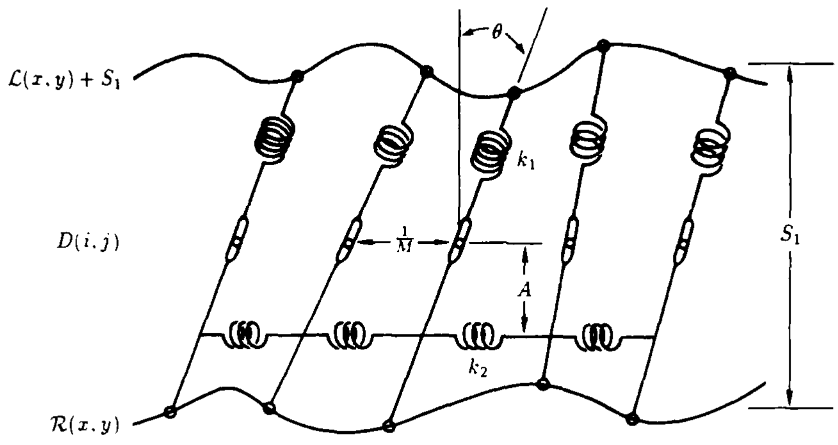
If we assume that $\mathcal{L}$ and $\mathcal{R}$ are commensurate, the first term in the integrand represents the photometric error associated with $\mathcal{D}$. The second term is the first-order variation of $\mathcal{D}$, or a measure $\mathcal{D}$'s "flatness." By minimizing $\mathcal{E}$ with respect to $\mathcal{D}$, therefore, we should find the simplest disparity map (in the sense of flattest) that adequately explains the image data.

Notice that disparity is a scalar field. Corresponding points may have different x coordinates, but they will always have the same y coordinate. This is a common assumption and involves no loss of generality: if the relative positions and orientations of the two cameras are known, as well as the internal camera parameters, correspondences are restricted to a family of epipolar lines. If the epipolar lines are not horizontal the images can easily be mapped into a normal stereo pair in which they are. We can write the 3D coordinates of the scene in the coordinate frame of the left camera as:

$$\mathrm{p}(x, y) = \frac{B}{\mathcal{D}}(x - \frac{\mathcal{D}}{2}, y, f),$$

where B is the baseline separation and f is the focal length.

Because we will refer to $\mathcal{E}$ as the energy of our system, it is helpful to have a picture of why this interpretation makes sense. One can readily see [1] that $\mathcal{E}$ corresponds to the potential energy of a system of coupled springs illustrated in one dimension in Figure 1.



vertical springs: spring constant k_1, rest length S_1

horizontal springs: spring constant k_2, rest length $S_2 = \frac{1}{M}$

Figure 1: A spring model

The model consists of two surfaces, $\mathcal{R}(x,y)$ below and $\mathcal{L}(x,y)+S_1$ above. Midway between these surfaces is a lattice of pivot points, and at each such point is an elastic lever arm, with rest length S_1 and spring constant k_1. The lever arms are free to rotate in the (x,z) plane (i.e., in epipolar planes), while their endpoints are constrained to lie on the two surfaces. The lever arms are connected to their neighbors by other springs with spring constant k_2 that exert torques over moment arm A. The angles of the lever arms represent disparity on an M^2 cyclopean lattice.

It is easy to show that the energy of this system is proportional to $\mathcal{E}$, with

$$\lambda = \left(\frac{k_2}{k_1}\right)\left(\frac{A}{S_1}\right)^2 .$$

Approaches to Minimizing E

Minimizing $\mathcal{E}$ directly is difficult because it is nonlinear. Witkin et al. described a method for optimizing a generalization of Eq. (1) that is essentially a sophisticated form of gradient descent that tracks the solution over increasingly finer scales [3]. The hope is that $\mathcal{E}$ is convex at a coarse scale and that relatively coarse intermediate solutions will place the system in the correct convex region at finer scales. They report that the method is prone to error when it encounters bifurcations in its trajectory. As the scale becomes finer, the system must "choose" which path to follow, and it cannot recover from a mistake because $\mathcal{E}$ may never increase. The solution is therefore critically dependent on initial conditions.

Another approach would be to simulate the dynamics of the spring model. A physical realization of the spring model would be a dynamic system of oscillators that would follow a trajectory through a $2M^2$ dimensional phase space. (Each lever arm has two degrees of freedom: θ and $\dot{\theta}$.) We could flesh out this model by specifying the moments of inertia and damping coefficients of the lever arms. We could also add a periodic forcing function to add energy to the system, balancing the energy dissipated by damping. Having done this, we could could write the differential equations of motion describing the model's deterministic dynamic behavior. In principle, we could trace the trajectory of the system through its phase space, gradually reducing the amplitude of the forcing function while keeping the system in dynamic equilibrium. There is little point in simulating the dynamics in such detail, however, because we know that even low-dimensional forced oscillators have chaotic attractors [4]. The dynamics will be effectively stochastic.

The remainder of this paper describes an alternative and much less expensive approach. Instead of modeling the full dynamics of the system, it models only the thermodynamics. Kinetic energy is modeled as heat.

A Discrete Model

At this point, we will discretize the problem by defining the lattices D, L, and R on $\mathcal{D}$, $\mathcal{L}$, and $\mathcal{R}$. D now has integer values and is interpreted as:

$$L_{i-\lfloor \frac{D_{i,j}}{2} \rfloor ,j} \text{ corresponds to } R_{i+\lceil \frac{D_{i,j}}{2} \rceil ,j} \ .$$

Eq. 1 becomes

$$E = \sum_{i,j} \{ [L_{i-\lfloor \frac{D_{i,j}}{2} \rfloor ,j} - R_{i+\lceil \frac{D_{i,j}}{2} \rceil ,j}]^2 + \lambda [\Delta D_{i,j}]^2 \ \} \tag{2}$$

where

$$\Delta D_{i,j} = \sqrt{\sum_{k,l \in \mathcal{N}_{i,j}} (D_{i,j} - D_{k,l})^2} \ .$$

$\mathcal{N}_{i,j}$ denotes the four nearest neighbors of (i,j). In terms of the spring model, the ends of the lever arms are now constrained to lie on a finite number of positions on the two surfaces.

Scaling

Disparity scales linearly with the size of the image. This suggests that a stereo matching system can begin its search at a coarse scale, find an approximate result, use this result to initialize its search at a finer scale, and so on. This has two benefits: it improves the efficiency of search by allowing the system to work initially in smaller phase spaces, and it reduces the false target problem by using a range of spatial scales. Large features are first detected at coarse scales, and their locations in finer scales are constrained.

We can construct a sequence of n lattices, $\{D^k\}$ for $k = 0, n-1$, which represent disparity maps of increasing precision, defined by the following rule:

$$D^k{}_{i,j} \Rightarrow L^k{}_{i-2^{n-k-1}\lfloor \frac{D_{i,j}}{2} \rfloor ,j} \text{ corresponds to } R^k{}_{i+2^{n-k-1}\lceil \frac{D_{i,j}}{2} \rceil ,j} \ . \tag{3}$$

Suppose the maximum range of disparity between a pair of images is $[-2^{n-1}+1, 2^{n-1}]$. The D^0 that matches these images must be binary, and it should be relatively easy to find the best D^0 because the phase space is relatively small. We can then use $2D^k$ as the initial condition of a search for D^{k+1}, until finally D^{n-1} will match the images with single-pixel precision.

When using this scaling method it is necessary to filter L and R to avoid aliasing. We use a difference-of-Gaussians (DOG) approximation to the Laplacian of a Gaussian. This transform can be computed efficiently by recursively applying a small generating kernel [5] to create a low-pass Gaussian sequence, and then differencing successive low-pass images to construct the bandpass DOG sequence. Low-pass filtering alone is adequate to avoid aliasing, but the bandpass filtering is useful for eliminating low-frequency error.

Stochastic Optimization

Standard (Canonical) Annealing

Simulated annealing is a fairly new technique for solving combinatorial optimization problems. The next section presents a new variety of simulated annealing (called microcanonical annealing) that has several advantages for computer implementation. In this section the basic principles of the standard form of simulated annealing are described to set a context for the introduction of microcanonical annealing.

The most fundamental result of statistical physics is the Boltzmann (or Gibbs) distribution

$$P_i = \frac{\exp(-E_i/kT)}{\sum_\nu \exp(-E_\nu/kT)} \ ,$$

which gives the probability of finding a system in state i with energy E_i, assuming that the system is in equilibrium with a large heat bath at temperature kT. (The constant k (Boltzmann's constant) converts temperature to units of energy. In the following discussion we will assume that $T = kT$.) The normalizing quantity in the denominator, called the partition function, is a sum over all accessible states.

Physicists would like to be able to calculate macroscopic equlibrium properties of model systems. In 1953 Metropolis et al. [6] described a Monte Carlo algorithm that generates a sequence of states that converges to the Boltzmann distribution in the limit. This method, which simulates the effect of allowing the system to interact with a much larger heat bath, samples what is called the canonical ensemble. Macroscopic parameters can then be calculated without knowledge of the partition function by averaging over long sequences.

The Metropolis algorithm begins in an arbitrary state and then successively generates candidate state transitions $(\nu - \nu')$ at random. A transition is accepted with the following probability:

$$Pr(\nu - \nu'|\nu, \nu') = \begin{cases} 1 & \text{if } \Delta E < 0 \\ \exp(-\Delta E/T) & \text{otherwise} \end{cases} \tag{4}$$

where $\Delta E = E_{\nu'} - E_{\nu}$. Asymptotic convergence of the Metropolis algorithm to the Boltzmann distribution is guaranteed if the process for generating candidate state transitions is ergodic.

Kirkpatrick et al. [7] and Černy [8] independently recognized a connection between the Metropolis technique and combinatorial optimization problems. If the energy of a state is considered as an objective function to be minimized, the minimum can be approximated by generating sequences at decreasing temperatures, until finally a ground state, or a state with energy very close to a ground state, is reached at $T = 0$. This is analogous to the physical process of annealing.

There are results showing the existence of annealing schedules (i.e., the rate of decrease of temperature) that guarantee convergence to ground states in finite time [9], but these schedules are too slow for practical use. Faster *ad hoc* schedules have been used in many problems with good average-case performance. While these faster schedules may not find an optimal state, they can converge to states that are very close to optimal.

Microcanonical Annealing

Creutz [10] has described an interesting alternative to the Metropolis algorithm. Instead of simulating the effect of a large heat bath, the Creutz algorithm simulates a thermally isolated system in which energy is conserved. Samples are drawn from the microcanonical ensemble. One can imagine the difference between the Metropolis algorithm and the Creutz algorithm as follows. The Metropolis algorithm generates a "cloud" of states, each with, in general, different energies, which fills a volume of phase space. As temperature decreases this volume contracts to one or more ground states. The Creutz algorithm, by contrast, generates states on a constant-energy surface in a somewhat larger phase space. As energy decreases these surfaces shrink to the same set of ground states.

The simplest way to accomplish this is to augment the system with one additional degree of freedom, called a demon, which carries a variable amount of energy, E_D. This demon holds the kinetic energy of the system and, in effect, replaces the heat bath. The total energy of the system is now

$$\begin{aligned} E_{total} &= E_{potential} + E_{kinetic} \\ &= E + E_D \end{aligned}$$

The demon energy, being kinetic, is constrained to be nonnegative. The algorithm accepts all transitions to lower energy states, adding $-\Delta E$ (the energy given up) to E_D. Transitions to higher energy are accepted only when $\Delta E < E_D$, and the energy gained is taken away from E_D. Total energy remains constant.

Microcanonical annealing simply replaces the Metropolis algorithm with the Creutz algorithm. Instead of explicitly reducing temperature, the microcanonical annealing algorithm reduces energy by gradually lowering the value of E_D. Standard arguments can be used to show that at equilibrium E_D assumes a Boltzmann distribution over time [10]:

$$Pr(E_D = E) \propto \exp(-E/T) \ .$$

Temperature therefore emerges as a statistical feature of the system:

$$T = \langle E_D \rangle \ . \tag{5}$$

Microcanonical annealing has several advantages over standard annealing:

- It does not require the evaluation of the transcendental function $\exp(x)$. Of course, in practice this function can be stored in a table, but we would like our algorithm to be suited to fine-grained systems with very limited local memory, like the Connection Machine.

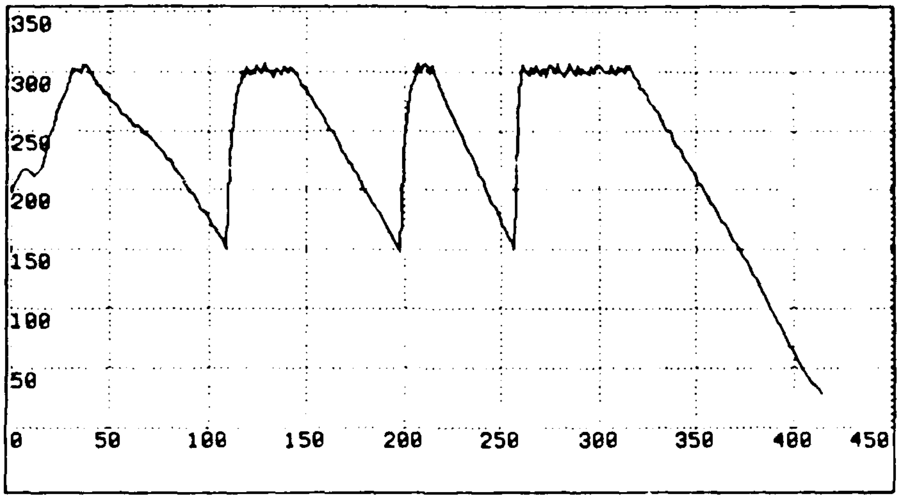


Figure 2: $T = \langle E_D \rangle$

- It is easily implemented with low-precision integer arithmetic — again, a significant advantage for simple hardware implementation.
- In the Metropolis algorithm a state transition is accepted or rejected by comparing $\exp(-\Delta E/T)$ to a random number drawn from a uniform distribution over $[0,1]$, and these numbers should be accurate to high precision. The Creutz algorithm does not require high-quality random numbers.

Implementation Details

The program is implemented in Release 5.0 *LISP and is fully compiled with the *LISP compiler. The Connection Machine at SRI is one-eighth of a full machine (8K vs. 64K processors). We have two Symbolics Lisp Machine front ends.

The input to the program is two images, L and R, and a number n that specifies how many levels of scaling are used. The images are assumed to satisfy the horizontal-epipolar condition and to have dimensions of the form $(2^M, 2^N)$, $M, N > 6$. The Connection Machine is assumed to be booted into the same configuration.[3]

The program then DOG filters L and R to create the sequences of bandpassed images $\{L^k\}$, $\{R^k\}$ for $k = 0, n-1$.

Next, beginning with zero disparity at level 0, the program executes a series of n heating and cooling cycles, using the result at level $k-1$ to initialize D^k, generating a sequence of states:

$$(S_0^0 \cdots S_{l_0}^0) \cdot \cdot (S_0^{n-1} \cdots S_{l_{n-1}}^{n-1}).$$

A transition between levels amounts to doubling the disparity values and updating the rule for interpreting D (see Eq. 3).

The value of $T \approx \langle E_D \rangle$ of each state of a typical run is plotted in Figure 2. Notice that this run has four levels of scaling. We raise the temperature to $T = 300$ by successively adding 10 units to each demon on each iteration. The system is allowed to dwell at $T = 300$ for awhile, and is then cooled by removing three units from each demon on each iteration. Except for the last cycle, cooling below about $T = 150$ is wasted effort because the fine structure "frozen in" below this temperature will be destroyed in the next heating cycle.

Let r_{eq} be the ratio of the observed average demon energy to the standard deviation of the same observed distribution:

$$r_{eq} = \frac{\langle E_D \rangle}{\sigma(E_D)} . \tag{6}$$

At equilibrium E_D will have a Boltzmann distribution, which implies that $r_{eq} = 1$. Figure 3 shows a plot of r_{eq} for

[3] The Connection Machine allows the user to have more than one configuration simultaneously, but this feature is not required here.

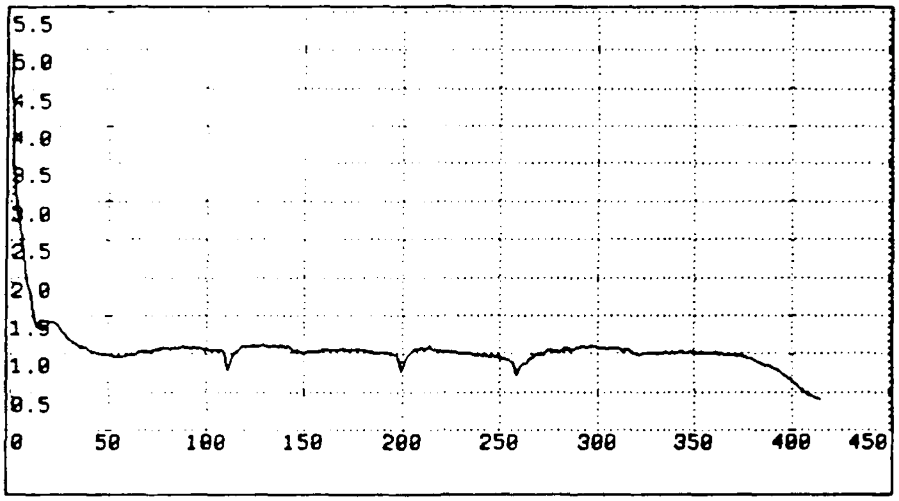


Figure 3: $r_{eq} = \frac{\langle E_D \rangle}{\sigma(E_D)}$

the same run as in Figure 2. Note that the plot of r_{eq} indicates that the system moves away from equilibrium during the relatively fast heating cycles, but relaxes quickly back to equilibrium after cooling starts. The system appears to drop away from equilibrium at low temperatures according to the r_{eq} plot, but this effect is actually because there are relatively few energy levels available to the demons near the ground state.

Choosing parameters of this procedure — heating and cooling rates, termination conditions, and so on — remains an art, as in virtually all applications of simulated annealing. The results in Section were generated with a common parameter set that was determined empirically from tests on a wide variety of images. A value of $\lambda = 64$ works well for typical images quantized into 8 bits.

As with the Metropolis algorithm, the Creutz algorithm converges to the Boltzmann distribution in the limit for any ergodic process generating candidate state transitions. Of course, different state-transition schemes will affect the rate of convergence. We have found the following simple method to be adequate:

$$P(d \rightarrow d') = \begin{cases} .5 & \text{if } |d - d'| = 1. \\ 0 & \text{otherwise.} \end{cases}$$

In other words, the disparities increase or decrease by one lattice position, or remain unchanged if the transition is rejected, as the system follows a Brownian path on its phase-space surface of constant energy.

Boundary conditions can be troublesome. Nonzero disparities near the edge of the lattice can match image points off the lattice. When this occurs we assign the photometric term in Eq. 2 a value equal to the current temperature, effectively placing an energy barrier at the boundary.

Annealing in Parallel

The essential inner loop of the algorithm, called a "full-pass", tries exactly one random transition for each lattice site. It is important to realize that we cannot update *all* sites in parallel. One full-pass should leave the total energy $E + E_D$ unchanged, but this is not ensured if two neighbors are updated simultaneously. This presents no problem for four-neighbor interactions: the lattice can be split into two "checkerboard" subsets that are updated sequentially. More complex neighborhoods would require more subsets, reducing parallelism.

The basic version of microcanonical annealing, using only one demon, is not suited to a parallel implementation. Each decision to accept or reject a state transition depends on the value of E_D and, therefore, on the previous decision. Instead, we use a lattice of demons. Temperature is still measured with Eq. 5, but using the distribution of E_D over space rather than time. Statistics can be sampled over both time and space, if desired.

There is a minor complication in using a lattice of demons. The single-demon algorithm visits sites at random and the demon allows energy to be transferred throughout the lattice. Similarly, in the lattice-of-demons algorithm the demons must be mixed throughout the lattice. We use a complete random permutation of the demons after every lattice update, but more local methods are also adequate.

lattice	computer	VP ratio	time	factor-speedup
128^2	SLM	n a	19 4 sec	n a
	4K	4	.28 sec	69
	8K	2	.26 sec	74
256^2	SLM	n a	76 7 sec	n a
	4k	8	54 sec.	142
	8k	4	36 sec	213

Table 1: Symbolics 3600 vs. Connection Machine

Experimental Results

This section presents experimental results for two distinct cases: an aerial stereo pair (Figure 4) and a ground-level scene with prominent occlusions (Figure 5).

The figures show the two original images (both are 128 x 128) and the disparity map at the end of each cooling cycle. Each example required about 4 minutes of processing time.

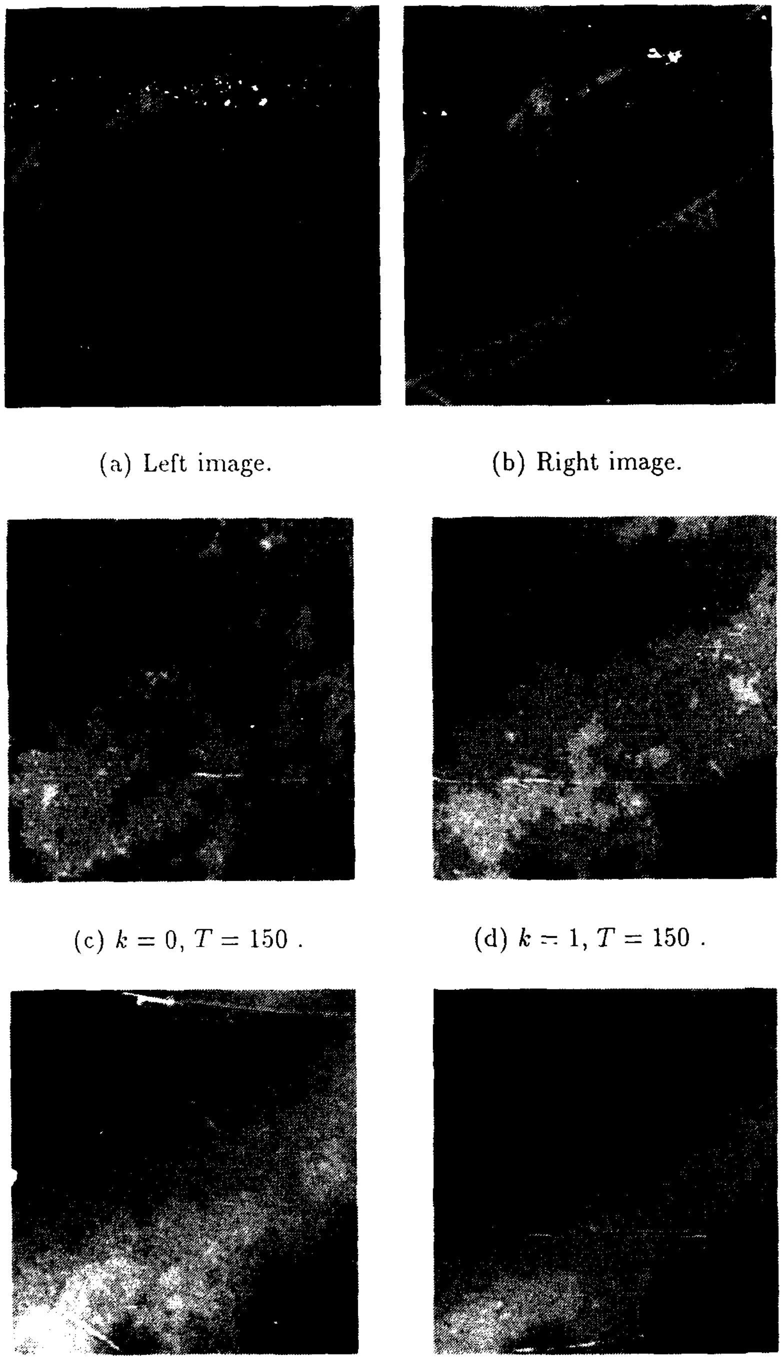
Table 1 indicates how different Connection Machine configurations compare to a Symbolics 3600 Lisp Machine. The timings are for one full-pass. Note that the efficiency of the Connection Machine depends strongly on the VP ratio, which is the ratio of the number of virtual processors to physical processors. This is because the overhead of the front end is too large to keep up with the Connection Machine at low VP ratios.

Conclusions

The method fits the Connection Machine architecture very well and was quite easy to implement. The processing is still too slow for real-time applications, but is adequate for cartography. By using a new feature of the Connection Machine software that allows the user to define virtual processor sets of different sizes, the implementation will be able to process $1K$ x $1K$ images in a reasonable time.

The use of a scale hierarchy dramatically increases the efficiency of the method, especially for large problems such as those illustrated in Figures 4 and 5. An additional benefit of using a scale hierarchy is that the solution is less sensitive to small amounts of vertical disparity, which is eliminated at coarser scales. (Uncertainty in the camera model will usually cause some vertical disparity in high-resolution images.) A Gaussian low-pass hierarchy works as well as the Laplacian hierarchy if the images are recorded with equivalent sensors. The benefit of bandpass filtering is to eliminate the low-frequency variation caused by uncalibrated photometry. Annealing provides a way to bridge the gap between scales. The microcanonical annealing algorithm appears to be an improvement over canonical annealing for reasons discussed in the section on that algorithm.

Canonical annealing and "pure" single-demon microcanonical annealing are at opposite ends of a spectrum. In canonical annealing the heat bath is much larger than the model system, and is not represented explicitly. In pure microcanonical annealing the heat bath — that is, the single demon — is much smaller than the system, and it is represented explicitly. The lattice-of-demons algorithm is midway between these extremes, with the heat bath and the model system having comparable sizes. In a sense, this is a classical space/time tradeoff. By representing the heat bath explicitly we can avoid the evaluation of complicated functions.



(a) Left image.

(b) Right image.

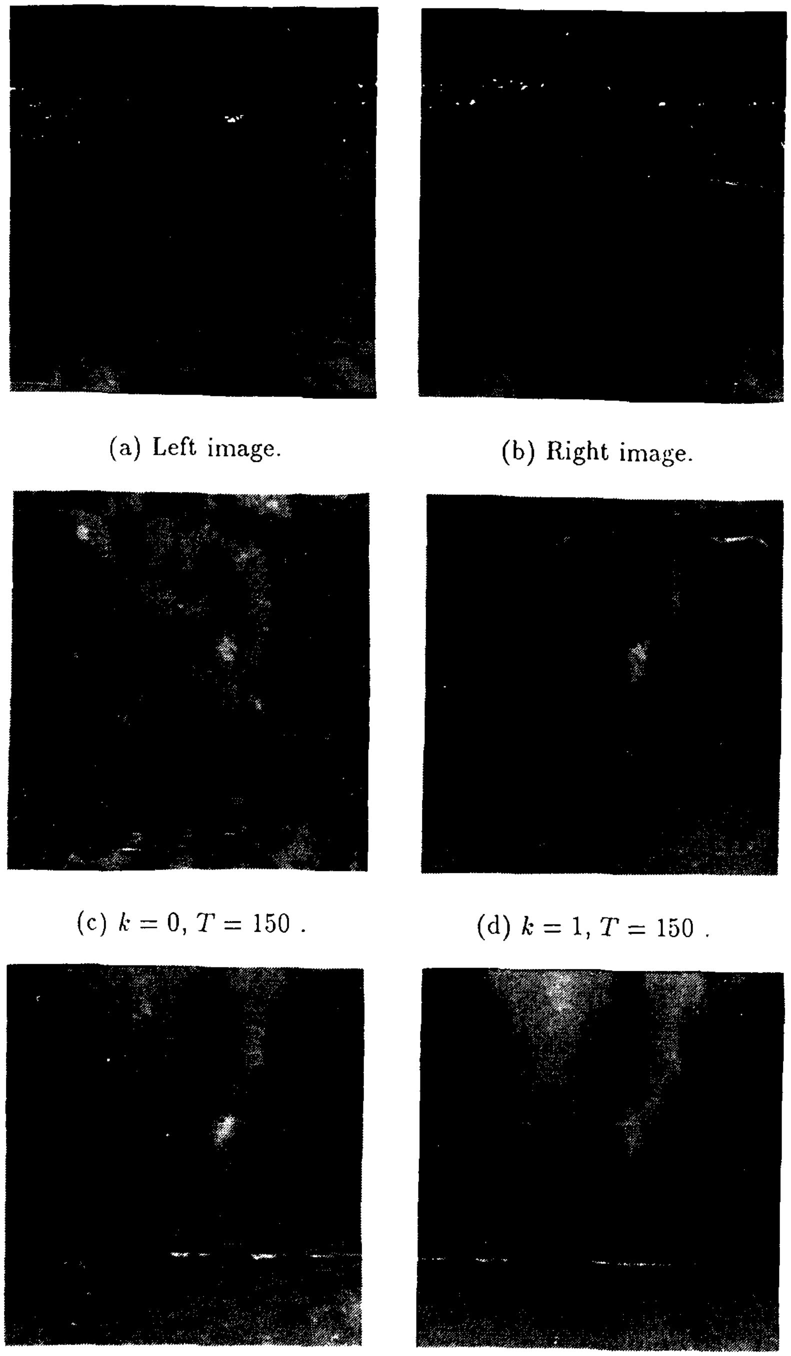
(c) $k = 0, T = 150$.

(d) $k = 1, T = 150$.

(e) $k = 2, T = 150$.

(f) $k = 3, T = 0$.

Figure 4: Aerial stereogram results



(a) Left image.

(b) Right image.

(c) $k = 0, T = 150$.

(d) $k = 1, T = 150$.

(e) $k = 2, T = 150$.

(f) $k = 3, T = 0$.

Figure 5: Ground-level stereogram results

References

[1] Barnard, S.T., "Stochastic Stereo Matching Over Scale," *International Journal of Computer Vision*, Vol. 2(4), 1988.

[2] Julesz, B., *Foundations of Cyclopean Perception*, University of Chicago Press, Chicago, IL, 1971.

[3] Witkin, A., D. Terzopoulos, and M. Kass, "Signal Matching Through Scale Space," *International Journal of Computer Vision*, Vol 1, pp.133-144, 1987.

[4] Walker, G.H., and J. Ford, "Amplitude Instability and Ergodic Behavior for Conservative Nonlinear Oscillator Systems," *Physics Review*, Vol. 188, pp.416-432, 1969.

[5] Burt, P., "The Laplacian Pyramid as a Compact Image Code," *IEEE Transactions on Communications*, Vol. COM-31, pp.532-540, 1983.

[6] Metropolis, N., A.W. Rosenbluth, M.N. Rosenbluth, A.H. Teller, and E. Teller, "Equations of State Calculations by Fast Computing Machines," *Journal of Chemical Physics*, Vol. 21, pp.1087-1092, 1953.

[7] Kirkpatrick, S., C.D. Gelatt, and M.P. Vecchi, "Optimization by Simulated Annealing," *Science,*, Vol. 220, pp.671-680, 1983.

[8] Černy, V., "Thermodynamical Approach to the Traveling Salesman Problem: An efficient Simulation Algorithm," *Journal of Optimization Theory and Applications*, Vol. 45, pp.41-51, 1985.

[9] Geman, S., and D. Geman, "Stochastic Relaxation, Gibbs Distributions, and Bayesian Restoration of Images," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol. 6, pp.721-741, 1984.

[10] Creutz, M., "Microcanonical Monte Carlo Simulation," *Physical Review Letters*, Vol. 50, pp.1411-1414, 1983.

Early Identification Of Occlusion In Stereo-Motion Image Sequences[1]

Poornima Balasubramanyam
Richard Weiss

Computer Vision Research Laboratory
Department Of Computer and Information Science
University Of Massachusetts
Amherst, Ma. 01003

Abstract

Detection and signalling of occlusion in motion and and stereo imagery has traditionally been based on approaches that involve the independent, full computation of flow and disparity in the respective image-pairs. These approaches tend to present a circular argument to the problem since prior knowledge of occlusion boundaries is essential for the prevention of smoothing across them in the computation of optic flow or disparity. In this paper, we use the information in both the stereo and motion sequences at two time instances to propose an early indicator of the presence of occlusion *prior* to the full computation of flow and disparity. Results are demonstrated on real stereo-motion imagery.

1.0 Introduction

1.1 Motivation

The research in the detection of occlusion boundaries corresponding to motion and depth discontinuities has traditionally not been a low level approach. Instead, the approach has been to attempt to infer such occlusion using the results of lower level processing such as optical flow computations or disparity computations.

One of the earliest approaches using flow information has been to determine the discontinuities in the magnitude of the flow created by an observer moving in a static environment [3]. Zero-crossings in the Laplacian of the computed optic flow field have also been used under the premise that sharp changes in the field signify discontinuities [10]. As with any approach that employs differentiation of the flow field, these approaches tend to be unstable under noise. Flow computation algorithms [1,4,5] tend to employ a global smoothness assumption in order to restore well-posedness to the problem of optic flow computation. Variations on this smoothness assumption range from the direct use of this constraint [4] to the less simplistic versions that allow for smoothing along the contour while inhibiting smoothing along the brightness gradient [5,7]. Confidence measures have also been devised to control the flow computation process [1]. The assumption of global smoothness, however, tends to result in an optic flow field that is erroneously smoothed across depth and motion discontinuities, and thereby confounding the later stages of occlusion boundary detection.

Similarly, in the problem of surface reconstruction using sparse disparity estimates, such as in [8], the explicit assumption has always been that the required surface vary as continuously as possible; even as smoothly as possible. This has the effect of erroneously smoothing over real world depth and orientation boundaries.

The problem, then, is one of a circular argument. The knowledge of the presence of discontinuities is very important to the correct flow and disparity computations especially at points in the image corresponding to object boundaries. On the other hand, how does one determine these discontinuities without flow (or disparity) already computed?

The field has been addressing these issues in recent years. It is accepted now that what is required is a means of computing both the flow (disparity) and the boundaries simultaneously. It is possible to use stochastic regularization techniques and propose line processes, using MRF methods, [6], to predict discontinuities in surface reconstruction. Similarly, Terzopolous [9], has addressed this issue by formulating multivariate non-quadratic stabilizers called controlled continuity stabilizers. In dealing with this in general visual reconstruction techniques, Blake and Zisserman [2] propose weak continuity constraints, i.e., continuity constraints are allowed to be broken in places where there is evidence of discontinuity. The big problem with all of these approaches is that the resulting variational functional becomes non-convex and any solution technique designed for the solution of variational functionals formulated with quadratic stabilizers, such as gradient descent, tends to get trapped in local minima. The graduated non-convexity algorithm of Blake and Zisserman [2] addresses this issue.

In [11],the problem of early detection of motion boundaries is tackled by defining five parametric and non-parametric statistical tests that can be conducted in a local neighborhood of the image point. The tests also

[1] This work has been funded in part by DARPA under grants: DACA76-85-C-0008; F30602-87-C-0140; DACA76-86-C-0015; and by NSF/CER under grant DCR-85003332

permit the simultaneous computation of flow along with the boundary computations and have been demonstrated on simulated imagery.

1.2 Our Approach

We are specifically interested in a reconstruction paradigm which can be categorized as "generalized stereo". The paradigm includes both conventional stereo approaches as well as the conventional motion approaches. Stereo imagery is a popular source for inferring depth of the world, while time-varying imagery is used to compute the motion of the environment relative to the camera system and indirectly compute depth. Extensive work has proceeded along both directions each independent of the other, and only recently attention is being focussed on an integrated approach to low level processing. Specifically with stereo and motion processing, the low level correspondence issues are very similar and similar geometric formalisms apply to both. Hence it seems strange that work in each field has proceeded without considering the possibility for integration of the two.

It should be possible to use the information in both the stereo and motion frames to simultaneously yield flow and disparity values together with the labelling of motion and depth discontinuities. Toward this end, we propose a method to determine the plausibility of locating a discontinuity at an image element from the stereo-motion frames.

In this paper we examine a proposed measure of confidence in the existence of a discontinuity at each image point from the stereo-motion pairs of images using potential flow and disparity estimates derived from hierarchical correlation processes. The results of using this measure to detect potential regions of occlusion boundaries corresponding to either motion occlusion or stereo occlusion are demonstrated. The results are shown for real imagery. The availability of such knowledge prior to the full computation of flow and disparity will undoubtedly be useful in guiding the smoothing processes in both flow and disparity computations as also surface reconstruction processes.

2.0 A Measure Of Occlusion In Stereo and Motion Imagery

In this section, a measure is defined to indicate the presence of occlusion in either stereo or motion imagery. A hierarchical algorithm is employed to control this processing from the coarser to the finer levels of resolution. Using the methodology in [1], this measure is applied hierarchically to the processing of flow and disparity fields.

2.1 The Flow-Disparity Estimate Error

Let us refer to the four stereo-motion intensity images as (L1, R1) and (L2, R2) for the left and right stereo frames at the first and second time instances, respectively (see Figure 1). In processing stereo-motion frames of intensity images, *in the absence of occlusion boundaries*, we can intuitively expect the following to hold true. In order to find a best match in R2 for an image point in L1, there are two possible orders for the computation. (As we are interested in detecting regions of occlusion, we do not consider the obvious direct matching between L1 and R2). First, the best match in L2 can be determined, followed by the best match in R2 for the match in L2. Second, the best match in R1 can be determined, followed by the best match in R2. In other words, there is an order dictated by processing motion first and stereo next, or stereo first and motion next. In the absence of occlusion or disocclusion for the point in all the four frames, the orders of the processing can be expected to be commutative. That is, the results from either order for determining a best match for a point in L1 can be expected to coincide within reasonable limits in R2. Figure 1 gives a visual representation of this.

To rewrite the above more precisely, we have the following: Consider a world point **W = (X,Y,Z)** that is visible in all the four frames of stereo and motion. If the image point has coordinates $w_{l1} = (x_{l1}, y_{l1})$ in image frame L1, let us call the best (flow) match estimate of this point in the image frame L2 pair as $w_{l2} = (x_{l2}, y_{l2})$, and the best (disparity) match estimate of this point in image frame R1 as $w_{r1} = (x_{r1}, y_{r1})$. Now consider the best (disparity) match estimate for the point w_{l2} in R2 and call it P. Similarly, consider the best (flow) match estimate for the point w_{r1} in the image frame R2 and call it Q. Denote the displacement of w_{l1} due to motion between the left motion frames as (Δ^l_x, Δ^l_y). Denote the displacement of w_{r1} due to motion between the right motion frames as (Δ^r_x, Δ^r_y). Denote the displacement of w_{l1} due to disparity between the first stereo frames as (δ^1_x, δ^1_y). Denote the displacement of w_{l2} due to disparity between the second stereo frames as (δ^2_x, δ^2_y).

Then, P and Q can be written in coordinate form as :

$$\begin{aligned} P &= [(x_{l1} + \Delta^l_x + \delta^2_x), (y_{l1} + \Delta^l_y + \delta^2_y)] \\ Q &= [(x_{l1} + \delta^1_x + \Delta^r_x), (y_{l1} + \delta^1_y + \Delta^r_y)] \end{aligned}$$

Now, there will be errors in determining the best matches in flow and disparity, i.e., errors in δ and Δ values, and P and Q will *not* exactly coincide even when occlusion or disocclusion is not present, (e.g., at points belonging to homogeneous regions). At textured and non-occluding points, where the local intensity surface is distinctive, the errors in the coincidences can be expected to exist but be reasonably small. Distinguishing errors (i.e., non-coinciding point locations of P and Q) due to occlusion from errors due to the usual low-level variations becomes critical. Following [1], the sum of squares difference (SSD) values between correlation windows around P and Q is used to distinguish between the two cases. The exception to this occurs when a occlusion boundary is flanked by very similar homogeneous occluding and occluded surfaces. Let us call the SSD value between P and Q for any point of interest w_{l1} in L1 as the *Flow Disparity Estimate Error (FDEE)* for that point.

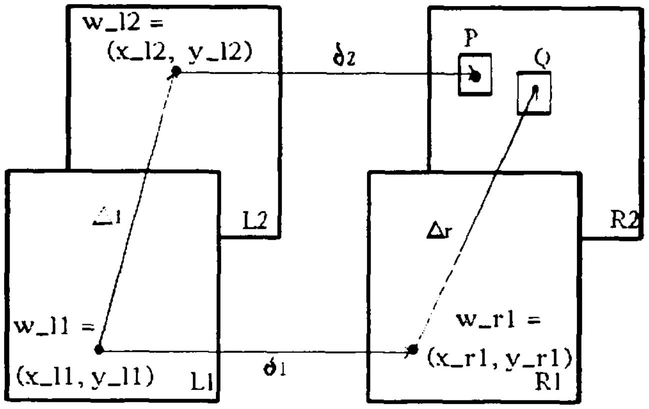


Figure 1 : The Flow-Disparity Error Estimate

2.2 The Hierarchical Correlation Algorithm

The FDEE computation uses the initial flow and disparity estimates computed from hierarchical SSD strategies. For the purposes of this paper, a dense computation of the FDEE is performed at every resolution level in a hierarchical representation of the intensity images, projecting the best match estimates at each level to the next higher level. The FDEE computation is done as follows:

1. Determine the Laplacian-filtered intensity images of all four stereo-motion intensity frames.
2. At each level from the *coarsest level* to the *finest level* loop

   a. For every image point in L1, find its best SSD match estimate in L2 and R1.

   b. For every image point in L2, find its best SSD match in R2.

   c. For every image point in R1, find its best SSD match in R2.

   d. Using steps **a**, **b**, **and c** determine for every point in L1, the FDEE value, using 5 X 5 gaussian weighted windows around the corresponding P and Q image coordinates

   e. Smooth the values of flow and disparity estimates from the above steps using the FDEE values as a mask to prevent smoothing across regions where the FDEE is high and project these smoothed flow and disparity estimates to the next higher level as initial estimates.

Notice that steps **a**, **b**, **and c** can be done in parallel. The FDEE computation is again a local parallel operation at every image point. It should be again emphasized that with such a scheme, the initial flow and disparity estimates are used at each level to compute the FDEE values for that level, and this is used as a mask in the smoothing formulation adopted in [1] at that level. The details of the Laplacian filtering techniques, the SSD match determination, the control strategy and the smoothing can all be seen in [1]. It is important to note that we use the FDEE as a mask in the smoothing at each level only for the current purpose of demonstrating its behavior near occlusion boundaries.

3.0 Experimental Results

The results of the FDEE computation have been shown for two sets of real stereo-motion sequences. The real greyscale stereo images are a data set from an indoor run with Moravec's "Neptune" vehicle at CMU[2]. The original (480 X 512) images were calibrated with an image warping algorithm developed by Moravec. The vehicle took 69 steps straight forward, each step being 3.866inches or 0.0981 meters. There are 70 stereo pairs including the pair taken before the first step. The physical set-up of the cameras was as follows: Height: 37 1/2 inches = 0.952 meters from floor; Left camera: 0.09 meters left of vehicle centerline; Right camera: 0.11 meters right of vehicle centerline; Overall baseline: 0.2 meters; The nominal focal points were 27 inches forward of the vehicle origin.

The FDEE computation is demontrated for two sets of stereo-motion sequences. For our purposes, the 480 X 512 images were reduced to a resolution of 128 X 128. The stereo images from the first and second steps of the vehicle form the data set for the first FDEE computation (see Figure 2). The coarsest resolution and finest resolution of computation were chosen to be (16 X 16) and (128 X 128), respectively. The final results at resolution level 7 (128 X 128) of the FDEE computation is shown in Figure 3, as a negative representation, with the darkest regions having the highest FDEE values. These results are in registration with the pixels at L1, as explained in Section 2.1.

In the image sequences, parts of the bookshelf are being occluded by the drum in the left stereo sequence. These regions correspond to high values of the FDEE. Notice that the region corresponding to the chalk marking on the floor is not marked with a high FDEE value along most of the marking because of an absence of occlusion.

[2] The information on the vehicle run and the camera setup are courtesy of Larry Matthies at CMU

On the other hand, the high FDEE values at the regions corresponding to the base and top of the robot vehicle are due to considerable intensity changes between the two sequences in these regions. This implies that, along with occlusions, large intensity changes such as ones that occur due to significantly expanding projections of surfaces between motion frames (due to looming of the objects), will also be signalled with high FDEE values. Similarly, regions of specularities (such as on the surface of the drum) tend to change in intensity values between stereo pairs as well as between motion pairs, and thus, they tend to get signalled as well. The occluding boundaries at the stand at the base of the drum and also the left occluding boundary of the drum are not marked with high FDEE values. This illustrates the case in which the occluding boundary is flanked by fairly homogeneous surfaces.

The stereo images from the 22nd and 23rd steps of the vehicle form the data set for the second FDEE computation (see Figure 4). As before, the results of the final FDEE computation at resolution (128 X 128) are shown in Figure 5 in negative format, and again, the results are in registration with the pixels at L1. As in the previous example the regions of occlusion and disocclusion are detected. Specularities are signalled as well as significantly expanding projections of surfaces that change in intensity between frames due to motion (looming).

4.0 Conclusion

This paper proposes the use of the information present in the four intensity images of a stereo-motion sequence to indicate the presence of occlusion in either of the stereo pairs or in either of the motion pairs or in both. It is clear that such a computation as the *Flow-Disparity Error Estimate* is by no means rigorously sufficient, since, for one thing, it is necessary to distinguish between boundaries due to the depth differences (and the resulting parallax) in the neighborhood and boundaries due to flow discontinuities in the absence of distinct depth differences. The latter case could occur when two objects are moving alongside one another with different motions. A factor we have not taken into consideration at all in the analysis presented here is the role that confidence measures (from the flow and disparity estimates [1]) could play in such computation. We are currently investigating the possibilities that arise from the integrated analysis of the FDEE with such confidence measures.

Another promising line of analysis is in the use of such occlusion markers in the continuous formulation for the computation of the flow and disparity values along with appropriate smoothing. Such a formulation can be used for surface reconstruction as well. Thus, in the terminology of [2], we are investigating the behaviour of an energy formulation in which a proposed discontinuity at a point is penalized, not by a constant penalty term, but by a term dictated via the results of occlusion detection preprocessors such as the FDEE.

Yet another area that needs to be investigated is the integration of such computation with the results of more sophisticated structures such as lines to distinguish between high FDEE values due to occlusions and those due to specularities or looming.

Acknowledgements: Many thanks to Ed Riseman for helpful comments and to Robert Heller for invaluable software support.

References

[1] Anandan, P."A Computational Framework and and Algorithm for the Measurement of Visual Motion," *IJCV*, Vol 2, pp 283-310, 1989.

[2] Blake, A. and Zisserman, A. "*Visual Reconstruction*," *The MIT Press*, 1987.

[3] Clocksin, W.F. "Perception of surface slant and edge labels from optical flow: A computational approach," *Perception* Vol.9, pp 253-269, 1980.

[4] Horn, B.K.P. " Determining Optical Flow," *Artificial Intelligence*, Vol.17, pp 185-203.

[5] Hildreth, E.C.," The Measurement Of Visual Motion," *Ph.D Dissertation* Dept. Of EE and CS, MIT, 1983.

[6] Marroquin, J. " Surface reconstruction preserving discontinuities," Memo 792, AI Laboratory, MIT, 1984.

[7] Nagel, H.H. and Enkelmann, W., " An investigation of smoothness constraints for the estimation of displacement vector fields from image sequences," *IEEE PAMI* Vol.8, 565-593, 1986.

[8] Terzopolous, D.,"Multiresolution computation of visible surface representation," *PhD Dissertation*, MIT, Jan. 1984.

[9] Terzopolous, D.," Regularization of inverse problems involving discontinuities,"*IEEE PAMI* Vol.8, 4, 413-424, 1986.

[10] Thompson, W.B., Mutch, K.M. and Berzins, V.A, " Dynamic Occlusion Analysis in Optical Flow Fields," *IEEE PAMI* Vol.7, no.4, 1985.

[11] Spoerri, A. and Ullman, S.," The Early Detection Of Motion Boundaries," *Proc. of ICCV*, 1987.

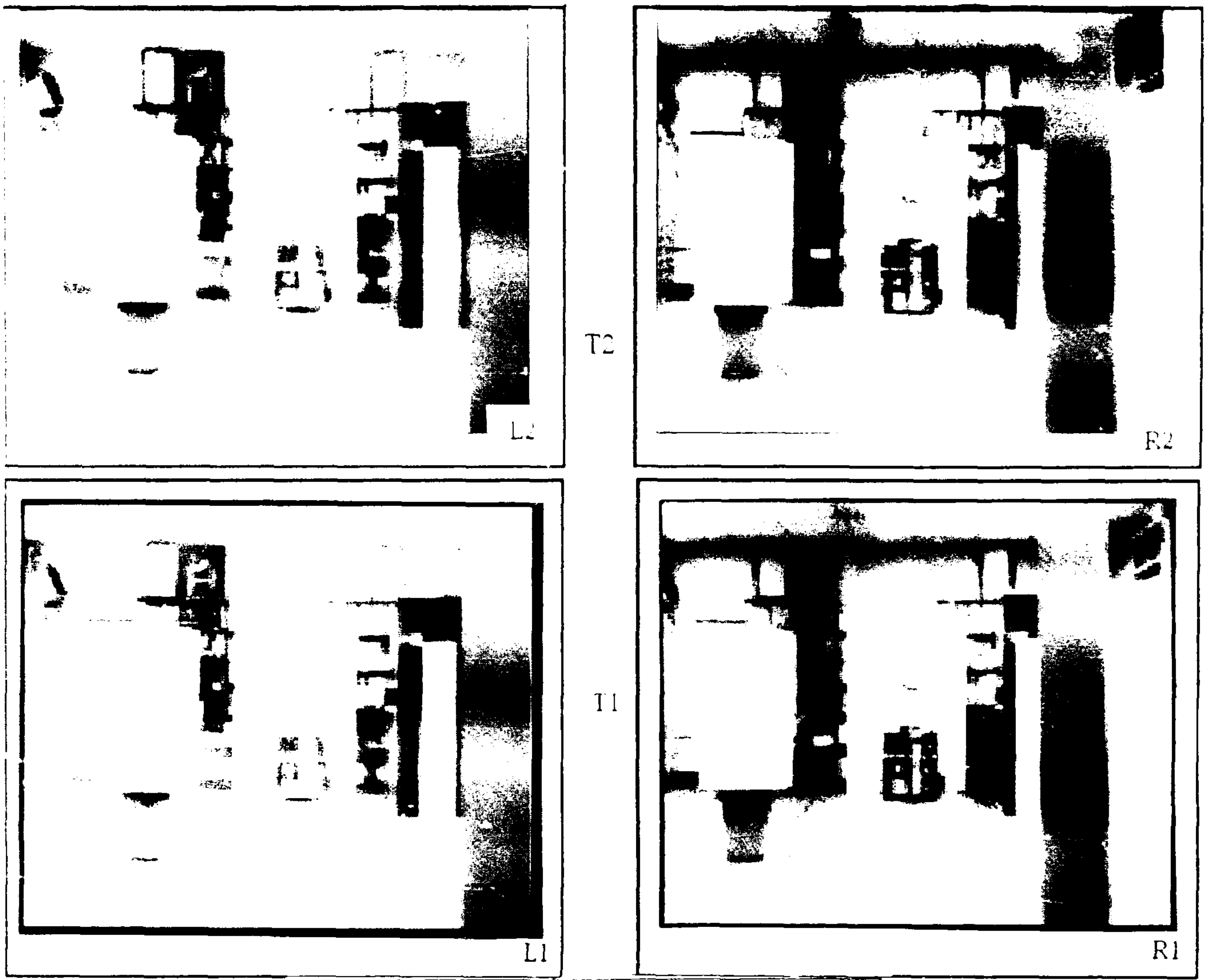


Figure 2 ↑

Stereo-Motion Sequence:
A step of 3.8 inches forward between T1 and T2.

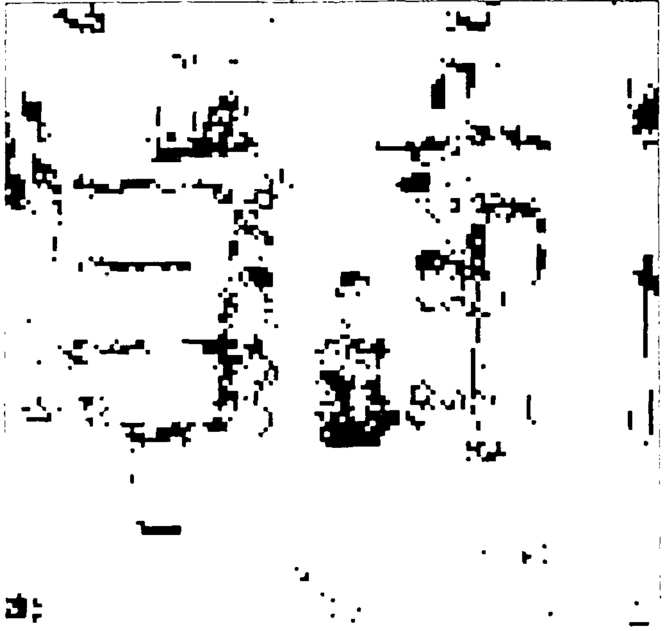


Figure 3

Stereo-Motion Sequence

The Flow Disparity Error Estimate is shown alongside

→

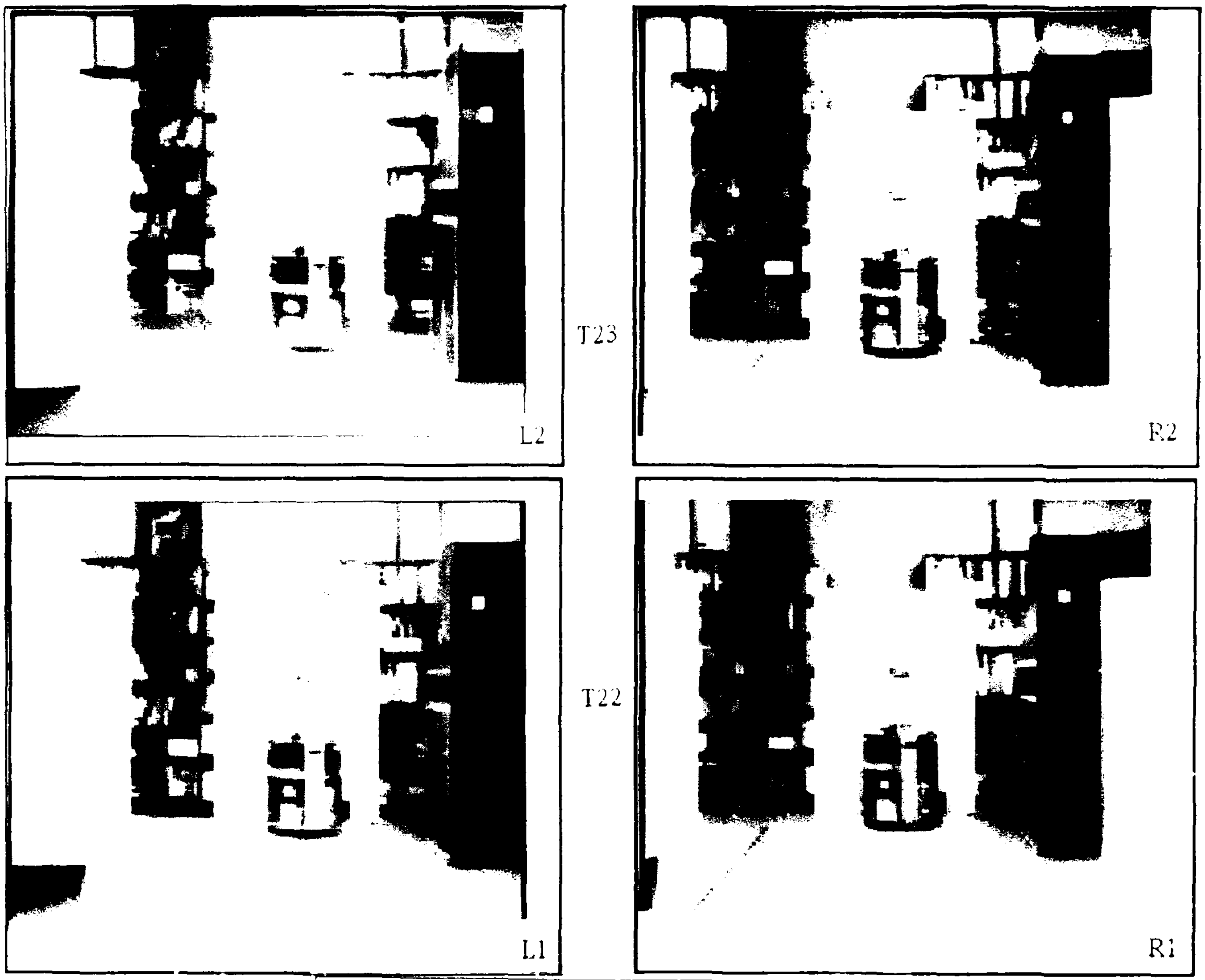


Figure 4 ↑

Stereo-Motion Sequence: A step of 3.8 inches forward between T22 and T23.

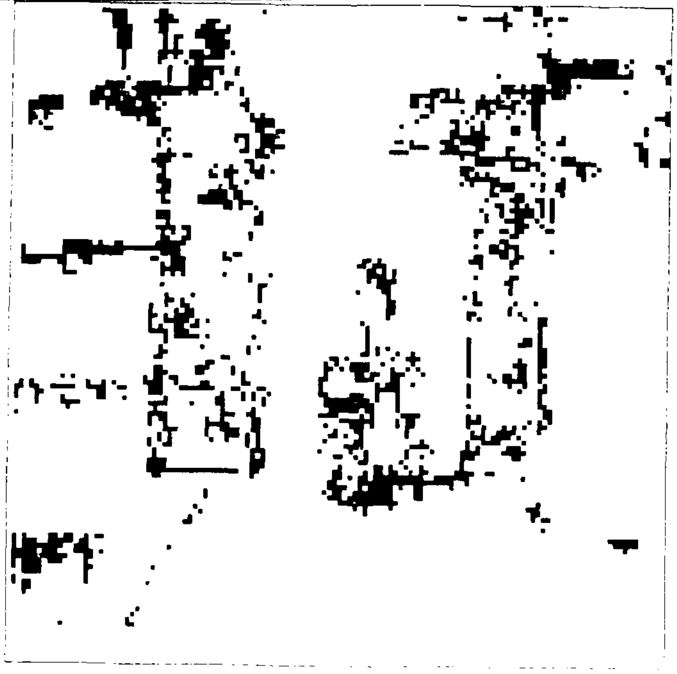
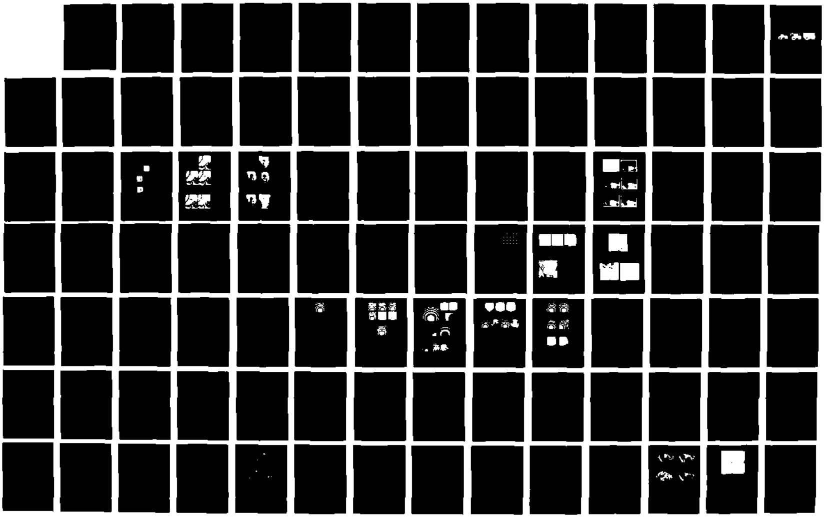


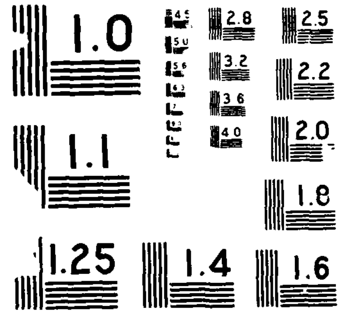
Figure 5

Stereo-Motion Sequence

The Flow Disparity Error Estimate is shown alongside →

AD-A212 981 PROCEEDINGS OF IMAGE UNDERSTANDING WORKSHOP HELD IN 12/12
PALO ALTO CALIFORNIA O. (U) SCIENCE APPLICATIONS
INTERNATIONAL CORP MCLEAN VA L S BAUMAN MAY 89
UNCLASSIFIED N00014-86-C-0700 F/G 17/11 NL




1.0
2.8
2.5
3.2
2.2
3.6
4.0
2.0
1.1
1.8
1.25
1.4
1.6

A PARALLEL COLOR ALGORITHM FOR SEGMENTING IMAGES OF 3-D SCENES

Glenn Healey

Robotics Laboratory
Department of Computer Science
Stanford University, CA 94305

ABSTRACT

This paper presents a parallel color algorithm for image segmentation. From an input color image, the algorithm labels each pixel in the image according to the corresponding material in the scene. This segmentation is useful for many visual tasks including inspection and object recognition. The segmentation algorithm is based on a detailed analysis of the physics underlying color image formation and may be applied to images of a wide range of materials. Surface material texture is dealt with in a natural way. An initial edge detection on the intensity image is used to guide the color segmentation process. The algorithm is inherently parallel and can be effectively mapped onto high performance parallel hardware.

INTRODUCTION

This paper develops an algorithm to segment images when little information is available about the objects or materials which might be present in the scene. In particular, I consider the problem of using a color image to determine the number of different materials in a scene and to label each image pixel as corresponding to one of these materials. Color information is essential to the segmentation algorithm because color allows the computation of image statistics which are independent of geometrical variations in the scene [3]. Therefore color, unlike image irradiance, can be used to group together regions of an image which correspond to the same material in the scene viewed under different geometrical conditions.

Researchers have proposed many different color segmentation techniques. One frequently used approach [11] is to use a clustering procedure [2] on the measured sensor values. Such a procedure attempts to find clusters of pixel values in color space and then assigns each image pixel to one of the clusters. Clustering techniques suffer from several disadvantages. In most real images, color clusters will overlap making pixel misclassification inevitable. Also, it is possible that surfaces of a single material in the scene which are viewed under different geometrical conditions will give rise to two different clusters in color space causing these surfaces to be assigned different labels.

Another approach to color segmentation is region splitting. Region splitting involves recursively breaking the input image into smaller and smaller pieces until each piece is uniform in some property. Ohlander [6] used histograms of feature values computed from a color image to guide region splitting. For each region this algorithm computes nine histograms from the input R, G, B values. The best peak in one of the histograms is used to threshold the region to separate out the pixels corresponding to the peak. Since histogram analysis is a global process, this method tends to miss small image regions which will not produce strong peaks in any histogram. Also, it seems unlikely that it is necessary to compute nine color features from the three measurements to generate a good peak in one feature histogram.

Ohta [7] uses Ohlander's algorithm to empirically determine if a small number of color features might be adequate for segmentation. Ohta determined from experiments on eight color images that an effective set of color features is given by $I_1 = (R + G + B)/3, I_2 = R - B$, and $I_3 = (2G - R - B)/2$ where I_1 is the most effective for segmentation and I_3 is the least effective. Though Ohta's analysis improves the Ohlander algorithm, it seems doubtful that eight images will be rich enough to lead to the determination of color features which are universally effective for segmentation. Another property of Ohta's color features is that the most significant color feature I_1 depends on the scene geometry. Applying Ohlander's algorithm with

Ohta's color features will cause curved surfaces of a single material to be broken into separate regions as in, for example, the segmentation of the color cylinder in [7].

More recently, Klinker [5] has derived a color segmentation algorithm from a physical model for reflection. Such an approach is more useful than previous techniques because it leads to algorithms which can account for the processes in the scene which contribute to image formation. Furthermore, it is easier to characterize what kinds of images an algorithm derived from physical models is likely to segment successfully. Klinker's algorithm is based on the dichromatic reflection model [9]. The algorithm segments the image into connected regions corresponding to objects of a single material. It is more sophisticated than previous segmentation techniques in that it is designed not to break the image because of highlights or because of intensity changes due to geometrical variation in the scene. This algorithm does, however, have several important limitations. Since the algorithm is derived from the dichromatic reflection model, it only applies to inhomogeneous dielectric materials. The algorithm uses a cylindrical model for clusters in color space which is unable to account for textural variations from material to material and for sensor noise properties which are not radially symmetric in color space. The current form of the algorithm does not attempt to identify separated image regions as instances of the same material in the scene.

THE SEGMENTATION PROBLEM

In [3] it is shown from general physical models that the reflectance of an opaque sample of material with one relevant interface can be accurately approximated by a function of the form

$$R(g,\lambda) = \begin{cases} M_S(g)C_S(\lambda) & \text{Metal} \\ M_S(g)C_S(\lambda) + M_B(g)C_B(\lambda) & \text{I.D.} \end{cases} \tag{1}$$

where g indicates dependence on the photometric angles and λ indicates dependence on wavelength. The subscript S denotes terms associated with surface (or interface) reflection and the subscript B indicates terms associated with body reflection. I.D. indicates an inhomogeneous dielectric material. The second part of (1) is equivalent to the dichromatic reflection model. I begin by assuming that the contribution of $M_S(g)$ to measured pixel values for I.D.'s is zero over most pixels in an image. This is a reasonable approximation for dielectrics over most values of the photometric angles. The approximation fails however at highlights and this failure will be dealt with later. Applying the approximation to (1) produces the result

$$R(g,\lambda) = M(g)C(\lambda) \tag{2}$$

which applies to both metals and inhomogeneous dielectrics.

Consider using N sensors with spectral sensitivities given by $f_i(\lambda)(0 \leq i \leq N-1)$ to obtain an image of a surface illuminated by a spectral power distribution $L(\lambda)$. At each point (x,y) in the image of the surface, measured sensor values s_i will be given by

$$\begin{aligned} s_i(x,y) &= \textstyle\int_\lambda f_i(\lambda)L(\lambda)M(g)C(\lambda)d\lambda & (3) \\ &= M(g)\textstyle\int_\lambda f_i(\lambda)L(\lambda)C(\lambda)d\lambda & (4) \end{aligned}$$

Thus for a material with reflectance given by (2) illuminated by a spectral power distribution $L(\lambda)$, the measured sensor values for that surface will lie on a line in sensor space which intersects the origin.

Several things might cause the sensor values measured for points on a surface to lie some distance from the line given by (4). Sensor noise will cause some scatter of points about the line. Variations in material composition from point to point on the surface will cause variations in the function $C(\lambda)$ which will also scatter points about the line. Such variations in material composition constitute one component of visual surface texture. The material discrimination problem is therefore equivalent to the problem of discriminating corrupted lines in N-dimensional sensor space. For most scenes, sensor noise and surface texture will cause lines in sensor space corresponding to different materials to overlap. Therefore it is important to characterize these lines accurately.

I begin by considering a simplified version of the segmentation problem. Suppose that it is known that there are M materials in the scene and certain sets of measured sensor vectors $S = (s_1, \cdots, s_N)$ are known to correspond to each material. We can model the distribution of pixels in sensor space for each material i using the multivariate normal density $p_i(S) = N(\mu_i, \Sigma_i)$

$$p_i(S) = \frac{1}{(2\pi)^{N/2}|\Sigma_i|^{1/2}} e^{\frac{-1}{2}(S-\overline{S}_i)^T \Sigma_i^{-1} (S-\overline{S}_i)} \tag{5}$$

where $\overline{S}_i$ is the n-element mean vector for material class i and Σ_i is the $n \times n$ covariance matrix. Suppose we wish to classify a new sensor vector v as belonging to one of the M material classes. To facilitate comparison which is not influenced by the effects of geometry in the scene, each distribution $p_i(S)$ is normalized to a multivariate normal distribution $\widehat{p}_i(S)$ described by

$$\widehat{\mu}_i = \frac{\mu_i}{||\overline{S}_i||} \tag{6}$$

$$\widehat{\Sigma}_i = \frac{\Sigma_i}{||\overline{S}_i||^2} \tag{7}$$

where $||\overline{S}_i||$ is the L^2 norm $||\overline{S}_i|| = \sqrt{s_1^2 + \cdots + s_N^2}$
Applying a similar normalization to v gives

$$\widehat{v} = \frac{v}{||v||} \tag{8}$$

The problem is to assign $\widehat{v}$ to one of the M material classes described by $N(\widehat{\mu}_i, \widehat{\Sigma}_i)$. From decision theory [2], the set of functions

$$g_i(\widehat{v}) = \log p(\widehat{v}/c_i) + \log P(c_i) \tag{9}$$

serve as discriminant functions for classification where c_i represents the ith material class. Thus, if we assign $\widehat{v}$ to the class c_i for which $g_i(\widehat{v})$ is the largest we will achieve minimum error rate classification. If all classes are equally likely, then $\log P(c_i)$ is independent of i and can be dropped from each discriminant function. Substituting the multivariate normal density into (9) gives

$$g_i(\widehat{v}) = \frac{-1}{2}(\widehat{v} - \widehat{\mu}_i)\widehat{\Sigma}_i^{-1}(\widehat{v} - \widehat{\mu}_i) - \frac{1}{2}\log|\Sigma_i| \tag{10}$$

The algorithm described in section 4 will use the discriminant functions of (10) during segmentation.

NORMALIZED COLOR

An important step described in section 2 is the computation of a normalized sensor vector

$$\widehat{S} = \frac{S}{\sqrt{s_1^2 + \cdots + s_N^2}} = (\widehat{s}_1, \cdots, \widehat{s}_N) \tag{11}$$

As suggested in section 2, the vector $\widehat{S}$ is independent of geometry because if in (4) we let

$$K_i = \int_\lambda f_i(\lambda)L(\lambda)C(\lambda)d\lambda \tag{12}$$

then

$$\widehat{s}_i(x,y) = \frac{M(g)K_i}{\sqrt{M(g)^2(K_1 + \cdots + K_N)}} = \frac{K_i}{\sqrt{K_1 + \cdots + K_N}} \tag{13}$$

and $\widehat{s}_i(x,y)$ depends on the sensors, the illuminant, and the reflecting material, but not on scene geometry.

Typically normalized color space coordinates (also called chromaticity coordinates) are generated using the L^1 norm giving the normalized coordinates of a sensor measurement s_i as $\frac{s_i}{s_1 + \cdots + s_N}$ [10],[4]. It is easy to show that as with the L^2 normalized coordinates, these normalized coordinates are independent of scene geometry. One property of this L^1 normalization is that for two distinct material lines in sensor space which are separated by a fixed angle, the distance between the normalized coordinates corresponding to these lines depends on the location of the lines in sensor space. Hence L^1 normalization has this anisotropic property when used for material discrimination. The L^2 normalization is not anisotropic in this sense and is therefore better suited for material discrimination.

It has been suggested on several occasions [2], [1], [8] that the normalized color of two points on a surface of the same material will be the same even if one of the points is directly illuminated and the other point is in a shadow. From (13), we see that this will not be true unless the light incident on the surface in the shadow has the same spectral power distribution $L(\lambda)$ as the direct illumination. In general, then, the normalized color corresponding to a point in a shadow will be different from the normalized color of a directly illuminated point of the same material on the same surface.

Kender [4] points out that the normalized color transformation has a nonremovable singularity at the zero signal point $s_1 = s_2 = \cdots = s_N = 0$ in sensor space and is highly unstable about this point. Also, sensor signal-to-noise is usually low near the zero signal point. For these reasons, it is advisable to consider measurements near the zero signal point too unreliable for a normalized color computation. This will prevent the system from hallucinating near the zero signal point.

THE SEGMENTATION ALGORITHM

A segmentation algorithm has been implemented which determines the number of materials in a color image and labels each pixel as an instance of one of the materials. Since, in general, a material discontinuity in the scene will cause an image irradiance discontinuity, the first step of the algorithm is to perform an edge detection on a single band image of the scene. This edge detection guides the color segmentation. Areas of the image which do not contain edge pixels are assumed to correspond to a single material in the scene.

The algorithm proceeds by examining increasingly smaller regions of the image. When execution begins, the entire image is placed on the region stack. An initially empty material list is maintained which, for each material, contains the current best estimate of $\widehat{\mu}_i$ and $\widehat{\Sigma}_i$ based on the pixels which have been assigned to material class i. When a region from the stack is processed, the region is first examined for edge pixels. If the region contains edge pixels it is split into subregions and the subregions are placed on the region stack. Currently, all regions are rectangular and, if necessary, are split into four rectangular subregions of equal size.

If a region contains no edge pixels, the region is assumed to correspond to a single material in the scene. For such a region R, a mean normalized color is computed. This normalized color vector v_R is analogous to the vector $\widehat{v}$ in section 2. The discriminant functions of (10) are computed for v_R for each class in the material list. If $g_i(v_R)$ is sufficiently large for some i, then the pixels in R are labeled class i and the material list statistics for class i are updated. If $g_i(v_R)$ is not sufficiently large for any class i, then R corresponds to a new material and a new material class based on R is added to the material list. When region size reaches one pixel, the edge test is eliminated and edge pixels are merged into existing material classes. Processing continues until the entire image is segmented. Certain precautions are taken during the segmentation. Pixels which fall near the zero signal point are labeled as being too dark to be classified reliably. When region size becomes very small, regions are merged into existing material classes rather than possibly becoming new classes.

The segmentation algorithm has several important properties. First, the algorithm begins by attempting to find large regions corresponding to a single material. If large regions are found, then highly representative statistics can be computed for material classes. Also, the algorithm will execute quickly on images of simple scenes. While the algorithm attempts to locate large regions, the dependence on edges ensures that good localization of region boundaries will be achieved. The algorithm is parallel since the bulk of the computing for each region can be done independently of any other part of the image.

EXPERIMENTAL RESULTS

The algorithm described in this paper has been successfully used to segment color images of several scenes containing a wide variety of different materials. Due to space limitations, I am unable to include examples of these results in the version of this paper in these Proceedings.

ACKNOWLEDGEMENTS

The author would like to thank Tom Binford for his help. This work has been supported in part by the Air Force Office of Scientific Research under contract F33615-85-C-5106 and by the Advanced Research Projects Agency of the Department of Defense under Knowledge Based Vision ADS subcontract S1093 S-1.

REFERENCES

1. M. Ali, W. Martin, and J. Aggarwal, "Color-based computer analysis of aerial photographs," *Computer Graphics and Image Processing*, 9, 282-293, 1979.

2. R. Duda and P. Hart, *Pattern Classification and Scene Analysis*, Wiley-Interscience, New York, 1973.

3. G. Healey, "Using Color for Geometry Insensitive Segmentation," *Journal of the Optical Society of America A*, June 1989.

4. J. Kender, "Saturation, Hue, and Normalized Color: Calculation, Digitization Effects, and Use," Carnegie-Mellon University Computer Science Department Technical Report, November 1976.

5. G. Klinker, S. Shafer, and T. Kanade, "Color Image Analysis with an Intrinsic Reflection Model," *Proceedings of Second International Conference on Computer Vision*, Tampa, 292-296, December 1988.

6. R. Ohlander, K. Price, and D.R. Reddy, "Picture Segmentation using a Recursive Region Splitting Method," *Computer Graphics and Image Processing*, Vol. 8, 1978, pp. 313-333.

7. Y. Ohta, T. Kanade, and T. Sakai, "Color Information for Region Segmentation," *Computer Graphics and Image Processing*, Vol. 13, 1980, pp. 222-241.

8. B. Schachter, L. Davis, and A. Rosenfeld, "Scene Segmentation by Cluster Detection in Color Space", University of Maryland Computer Science TR 424, 1975.

9. S. Shafer, "Using Color to Separate Reflection Components," COLOR Research and Application 10(4), 210-218 (1985). Also available as University of Rochester technical report TR 136, April 1984.

10. G. Wyszecki and W. Stiles, *Color Science: Concepts and Methods, Quantitative Data and Formulae*, 2nd edition, Wiley, New York, 1982.

11. R. Haralick and G. Kelly, "Pattern recognition with measurement space and spatial clustering for multiple images," *Proceedings of the IEEE*, 57, April 1969, 654-665.

ADAPTIVE IMAGE SEGMENTATION USING A GENETIC ALGORITHM

Bir Bhanu, Sungkee Lee, and John Ming

Honeywell Systems and Research Center
3660 Technology Drive
Minneapolis, MN 55418

ABSTRACT

Machine learning will play a critical role in future computer vision systems which must operate in dynamic out door scenarios. In this paper, we present a system which applies a machine learning technique to the problem of image segmentation. The learning technique, known as a *genetic algorithm*, allows the segmentation process to adapt to changes in image characteristics caused by variable environmental conditions such as time of day, time of year, clouds, rain, etc. The genetic algorithm efficiently searches the enormous hyperspace of segmentation parameter combinations using a collection of search points known as a population. By combining high performance members of the current population to produce better parameter combinations, the genetic algorithm is able to locate the parameter set which maximizes the segmentation quality criteria. We present some initial results which demonstrate the ability to adapt the segmentation parameters to variable lighting conditions (intensity and position of light sources).

1. INTRODUCTION

An innovative combination of techniques from two branches of science can often produce significant breakthroughs in the evolution of a technology. We have developed a system which applies a technique from the machine learning field to the computer vision problem of image segmentation. Image segmentation is typically the first, and most difficult, task of any automated image understanding process. All subsequent interpretation tasks, including feature extraction, object detection, and object recognition, rely heavily on the quality of the segmentation process. Despite the large number of segmentation techniques presently available,[2,8,14] no general methods have been found which perform adequately across a diverse set of imagery. Only after numerous modifications to an algorithm's control parameter set can any current method be used to process the wide diversity of images encountered in dynamic outdoor applications such as the operation of an autonomous robotic land/air vehicle, automatic target recognizer, or a photointerpretation task.

When presented with an image from one of these application domains, selecting the appropriate set of algorithm parameters is the key to effectively segmenting the image. The image segmentation problem can be characterized by several factors which make parameter selection process very difficult. First, most of the powerful segmentation techniques available today contain numerous control parameters which must be adjusted to obtain peak performance. The size of the parameter search space in these systems can be prohibitively large unless it is traversed in a highly efficient manner. Second, the parameters within most segmentation algorithms typically interact in a non-linear fashion, which makes it difficult or impossible to model their behavior in an algorithmic or rule-based fashion. Thus, the resulting objective function which results from various parameter combinations cannot generally be modeled in a mathematical way. Next, since variations between images cause changes in the segmentation results, the objective function varies from image to image. The search technique used to optimize the objective function must be able to adapt to these variations between images. Finally, the definition of the objective function itself can be subject to debate because there are no single, universally accepted measures of segmentation performance available with which to uniquely define the quality of the segmented image.

Hence, we need to apply a technique which can efficiently search the complex space of plausible parameter settings and locate the values which yield optimal results, without having to rely on knowledge of the particular application domain or detailed knowledge pertinent to the selected segmentation algorithm. Genetic algorithms, which are designed to efficiently locate the approximate global maxima in a search space, show great promise in solving this parameter selection problem.

The next section of this paper discusses the selection of the genetic algorithm as the appropriate search technique for this problem domain. Section 3 presents a brief overview of the details of the genetic process, including previous applications in computer vision research. Following this review, Section 4 describes the adaptive image segmentation process that we have developed. We explain the choice of a particular segmentation algorithm as well as the manner in which segmentation quality is measured. Section 5 provides some initial experimental results using the adaptive segmentation system. Finally, Section 6 discusses the conclusions of this work and describes future research plans.

2. SELECTION OF AN OPTIMIZATION TECHNIQUE

We previously highlighted some of the characteristics of the segmentation problem such as the size of the parameter search space, the complexity of the objective function, and variations in the objective function caused by changes in the imagery as well as the accepted definition of the function itself. Figure 1 provides a generalized representation of an objective function that is typical for the image segmentation process. The figure depicts a simplified application in which only two segmentation parameters are being varied, as indicated by the x and y axes. The z axis indicates the corresponding segmentation quality obtained for any pair of algorithm parameters. Because the algorithm parameters interact in complex ways, the objective function is multimodal and presents problems for many commonly used optimization techniques. Further, since the surface is derived from an analysis of real world imagery, it may be discontinuous, may contain significant amounts of noise, and can not be described in closed form.

The conclusion which can be drawn from Figure 1 is that we must identify a highly effective search strategy which can withstand the breadth of performance requirements necessary for the image segmentation task. We have reviewed many of the techniques commonly used for function optimization to determine their usefulness for this particular task. In addition, we have also investigated other knowledge-based techniques which attempt to modify segmentation parameters using production rule systems. The drawbacks to each of these methodologies are as follows:

- *Exhaustive Techniques (Random walk, depth first, breadth first, enumerative)* - Able to locate global maximum but computationally prohibitive because of the size of the search space.
- *Calculus-Based Techniques (Gradient methods, solving systems of equations)* - No closed form mathematical representation of the objective function is available. Discontinuities and other complexities present in the objective function.
- *Partial Knowledge Techniques (Hill climbing, beam search, best first, branch and bound, dynamic programming, A*)* - Hill climbing is plagued by the foothill, plateau, and ridge problems. Beam, best first, and A* searches have no available measure of goal distance. Branch and bound requires too many search points while dynamic programming suffers from the *curse of dimensionality.*
- *Knowledge-Based Techniques (Production rule systems, Heuristic methods)* - These systems have a limited domain of rule applicability, tend to be *brittle*, and are usually difficult to formulate. Further, the visual knowledge required by these systems may not be representable in knowledge-based formats.

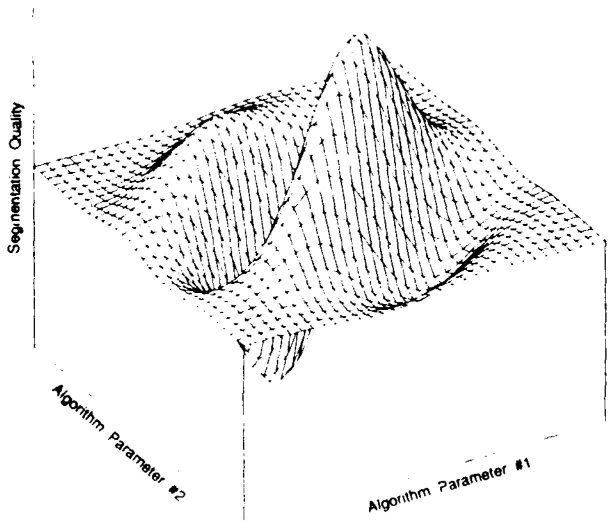


Figure 1: Representation of the objective function which must be optimized in the adaptive image segmentation problem.

Genetic algorithms are able to overcome many of the problems mentioned in the above optimization techniques. They search from a *population* of individuals (search points), which make them ideal candidates for parallel architecture implementation, and are far more efficient than exhaustive techniques. Since they use simple recombinations of existing high quality individuals and a method of measuring current performance, they do not require complex surface descriptions, domain specific knowledge, or measures of goal distance. Moreover, due to the generality of the genetic process, they are independent of the segmentation technique used, requiring only a measure of performance for any given parameter combination. Genetic algorithms are also related to simulated annealing[3] where, although random processes are also applied, the search method should not be considered directionless. In the image processing domain, Geman and Geman[9] have used simulated annealing to perform image restoration and Sontag and Sussmann[24] have performed image restoration and segmentation. Simulated annealing and other hybrid techniques[1] have the potential for improved performance over the earlier optimization techniques. However, in this paper, we will describe our experiments using genetic algorithms alone.

3. OVERVIEW OF GENETIC ALGORITHMS

Genetic algorithms were pioneered at the University of Michigan by John Holland.[4,15] The term *genetic algorithm* is derived from the fact that its operations are loosely based on the mechanics of genetic adaptation in biological systems. Genetic algorithms can be briefly characterized by three main concepts: a Darwinian notion of fitness or strength which determines an individuals likelihood of affecting future generations through reproduction; a reproduction operation which produces new individuals by combining selected members of the existing population; and genetic operators which create new offspring based on the structure of their parents.

A genetic algorithm maintains a constant-sized *population* of candidate solutions, known as individuals. The initial seed population can be chosen randomly or on the basis of heuristics, if available for a given application. At each iteration, known as a *generation*, each individual is evaluated and recombined with others on the basis of its overall quality or *fitness*. The expected number of times an individual is selected for recombination is proportional to its fitness relative to the rest of the population. Intuitively, the high strength individuals can be viewed as providers of "building blocks" from which new, higher strength offspring can be constructed.

The inherent power of a genetic algorithm lies in its ability to exploit, in a highly efficient manner, information about a large number of individuals. By allocating more reproductive occurrences to above average individuals, the overall net affect is an upward shift in the population's average fitness. Since the overall average moves upward over time, the genetic algorithm is a "global force" which shifts attention to productive regions (groups of highly fit individuals) in the search space. However, since the population is distributed throughout the search space, genetic search effectively minimizes the problem of converging to local maxima.

New individuals are created using two main genetic recombination operators known as *crossover* and *mutation*. Crossover operates by selecting a random location in the genetic string of the parents (crossover point) and concatenating the initial segment of one parent with the final segment of the second parent to create a new child. A second child is simultaneously generated using the remaining segments of the two parents. Mutation provides for occasional disturbances in the crossover operation by inverting one or more genetic elements during reproduction. This operation insures diversity in the genetic strings over long periods of time and prevents stagnation in the convergence of the optimization technique. The individuals in the population are typically represented using a binary notation to promote efficiency and application independence in the genetic operations. Holland[15] provides evidence that a binary coding of the genetic information may be the optimal representation. Other characteristics of the genetic operators remain implementation dependent, such as whether both of the new structures obtained from crossover are retained, whether the parents themselves survive, and which other structures are replaced if the population size is to remain constant. In addition, issues such as the size of the population, crossover rate, mutation rate, generation gap, and selection strategy have been shown to affect the efficiency with which a genetic algorithm operates.

Since they rely on the accumulation of evidence rather than on domain dependent knowledge, genetic algorithms are ideal for optimization in applications where domain theories or other applicable knowledge is difficult or impossible to formulate. However, there are certain drawbacks to genetic algorithms which make them inappropriate for certain applications. For example, genetic system usually require the evaluation of a large number of candidate solutions. In application domains where the evaluation process is expensive, the computational effort to perform numerous evaluations may be prohibitive. However, research by Fitzpatrick and Grefenstette[5] has shown that a simple statistical approximation to a complex evaluation process can allow genetic systems to effectively adapt in these situations and converge to global maxima.

To date, genetic algorithms have been applied to a wide diversity of problems. They have been used in combinatorial optimization,[13,22] VLSI layout,[7] gas pipeline operations,[11,12] and machine learning.[16,23] With regards to computer vision applications, Fitzpatrick et. al[6] have used genetic algorithms in solving the vision problem of image registration. In this work, the genetic system was used to select a set of transformation parameters which correctly align a pair of images. Genetic algorithms have also been used in computer vision for generating image domain feature detectors by Gillies.[10]

4. ADAPTIVE IMAGE SEGMENTATION

Genetic algorithms can be used in three different fashions to facilitate an adaptive behavior within a computer system. The simplest approach is to allow the genetic system to modify the set of control parameters which affect the output of an existing computer program. By monitoring the quality of the resulting program output, the genetic system can dynamically change the parameters to achieve the best performance. A second approach allows the genetic component to modify the complex data structures within an algorithm or production rule system. By modifying the control mechanism or *agenda* in an algorithm or the organization of data frames in a rule-based system, the genetic algorithm can bring about changes in the system's behavior. Finally, the most complex implementation allows the genetic system to actually make changes in the executable code of a program. In most ca^es, this adaptation involves changing the condition/action statements of a rule in a production system. Since almost every segmentation algorithm contains parameters which are used to control the segmentation results, we have adopted the first strategy listed above.

Adaptive image segmentation requires this ability to modify control parameters in order to respond to changes which occur in the image as a result of varying environmental conditions. The block diagram of our approach to adaptive image segmentation is shown in Figure 2. After acquiring an input image, the system analyzes the image characteristics and passes this information, in conjunction with the observed external variables, to the genetic learning component. Using this data, the genetic system selects an appropriate parameter combination, which is passed to the image segmentation process. After the image has been segmented, the results are evaluated and an appropriate reward is generated and passed back to the genetic algorithm. This process continues until a segmentation result of acceptable quality is produced using a set of control parameters. The details of each component in this procedure will be described in the following subsections.

4.1 IMAGE CHARACTERISTICS

The input image must be analyzed so that a set of features can be extracted to aid in the parameter selection process by the genetic component. A set of characteristics of the image is obtained by computing specific properties of the digital image itself as well as by observing the environmental conditions in which the image was acquired. Each type of information encapsulates knowledge that can be used to determine an appropriate starting point for the parameter adaptation process.

Image analysis produces a set of image statistics which measure various properties of the digital image. There are a large number of plausible image statistics which can be used, including:

- **First Order Properties:** Measure the shape of the first-order image histogram. Information includes mean, variance, skewness, kurtosis, energy, and entropy.
- **Second Order Properties:** Measure the histogram features based on joint probability distributions between pairs of pixels. Information includes autocorrelation, covariance, inertia, cooccurrence matrices, and other derived properties.

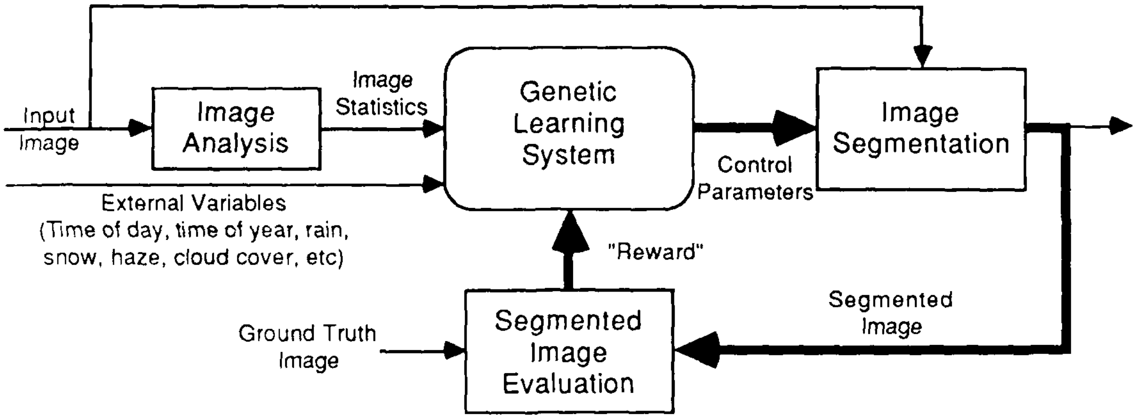


Figure 2: Block diagram of the adaptive image segmentation process.

- **Histogram Peak/Valley Properties:** Measure the values of the peaks and valleys in the image histogram. Information includes maximum peak height divided by minimum valley height, total number of histogram peaks, maximum peak location, minimum valley location, distance between maximum peak and minimum valley, and maximum peak-to-valley ratio.

For the purposes of our initial research, we have ignored the second order histogram properties since they are expensive to compute. Further, they may not be necessary if the selected first order properties and the peak/valley properties are sufficiently able to represent the internal image characteristics.

External variables can also be used to characterize an input image. These factors specify the conditions in which the image was acquired. They include information such as the time of day, time of year, cloud cover, temperature, humidity, and other environmental factors such as the presence of rain, snow, haze, fog, etc. These conditions all affect the quality of the image, which in turn necessitates changes in control parameters, and thus they provide useful information in representing the overall characteristics of the input image.

4.2 GENETIC LEARNING SYSTEM

Once the image statistics and external variables have been obtained, the genetic learning component uses this information to select an initial set of segmentation algorithm parameters. A simple classifier system is used to represent the image characteristics and the associated segmentation parameters. Figure 3 shows a simplified example of a classifier used by the genetic learning component. The classifier stores the current fitness of the parameter settings, the image statistics and external variables of the image, and the segmentation parameter set which is being adapted by the genetic algorithm. The image statistics and external variables form the condition portion of the classifier, C_1 through C_{I+J}, while the segmentation parameters indicate the actions, A_1 through A_N, of the classifier. Note that only the fitness value and the action portion of the classifier are subject to genetic adaptation; the conditions remain static for the life of the classifier.

The classifier used in our initial experiments is somewhat more complex than the one pictured in Figure 3 since we are using color imagery. We compute the first order histogram statistics and the histogram peak/valley properties for each of the red, green, and blue components of the color image. All of this information is then stored in the classifier. The external image variables, however, retain the same representation as shown in Figure 3.

Using the image characteristics for a new image, the genetic learning system compares this information with the current population of classifiers. The algorithm computes a ranked list of individuals which have characteristics similar to the current image. Using the highest ranked individual first, the genetic algorithm sends the parameter set from the selected individual to the segmentation component. After the image has been segmented, the results are reviewed by the evaluation system. If the segmentation quality is above a predefined threshold of acceptance, the process terminates and a new classifier is created using the characteristic values obtained from the new image, the parameter values which led to the acceptable results, and the fitness value that was achieved. The new classifier is added to the current population, replacing the individual which currently has the weakest fitness value.

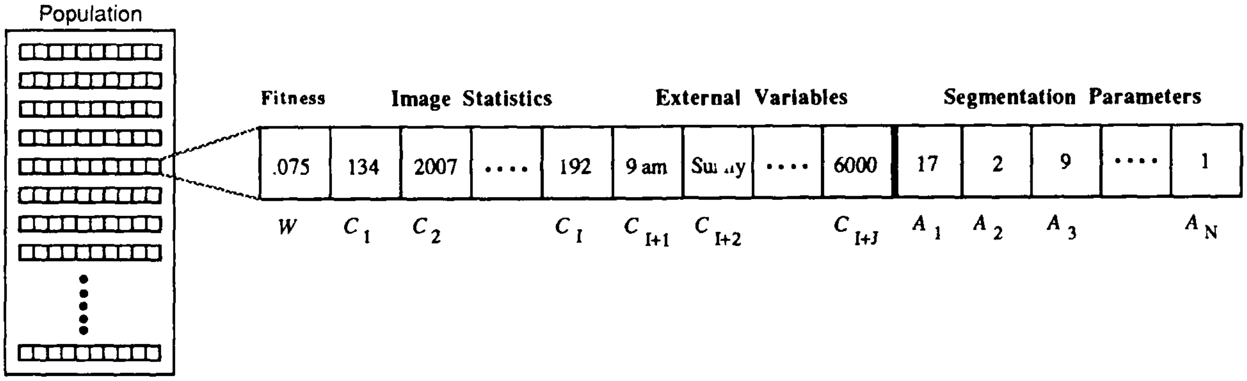


Figure 3: Representation of a classifier used by the genetic learning system.

Alternatively, if after testing some preset number of the ranked classifiers, the system has not achieved acceptable segmentation quality, the genetic learning process is invoked on the set of tested classifiers. A new seed population of classifiers is temporarily created using the characteristics from the current image in each of the condition fields and the parameters sets from each of the tested classifiers in the action field. The genetic process is then applied to the seed population until acceptable performance levels are achieved or a maximum number of segmentations have been performed. At this point, some portion from the set of new classifiers is allowed to replace the weakest elements in the initial population. We only allow a fraction of the new classifiers back into the original population to avoid skewing the population towards the characteristics of the most recently processed image. Once the least fit members of the population have been replaced, the system is ready to process a new image.

4.3 SEGMENTATION ALGORITHM

Since we will be working with color imagery in our experiments, we have selected the PHOENIX segmentation algorithm developed by Shafer and Kanade at Carnegie-Mellon University.[18,21] This system employs a region splitting technique[19] which uses information from the histograms of the red, green, and blue image components simultaneously. Basically, the algorithm recursively splits regions in the image into smaller subregions on the basis of a peak/valley analysis of the various color histograms. Fifteen different parameters[18] are used to control the thresholds and termination conditions used within the algorithm. Of these fifteen values, we have selected the two of the most critical parameters which affect the overall results of the segmentation process, *maxmin* and *hsmooth*. Maxmin specifies the lowest acceptable peak-to-valley-height ratio used when deciding whether or not to split a large region into two or more smaller parts. Hsmooth controls the width of the window used to smooth the histogram of each image region during segmentation. Smoothing helps to remove small histogram peaks corresponding to noise in the image. Future experiments may increase the number of selected parameters used for adaptation in order to investigate more difficult segmentation tasks.

From an analysis of the PHOENIX algorithm, we find that incorrect values in the two main parameters lead to results in which, at one extreme, the target in not extracted from the background, to the other extreme in which the target is broken up into many small regions that are of little use to higher level processes. By measuring segmentation performance using appropriate quality criteria, th; genetic process attempts to identify a parameter set that yields results between these two extremes. The segmentation quality criteria are described in the next section.

4.4 SEGMENTATION EVALUATION

After the image segmentation process has been completed by the PHOENIX algorithm, we must measure the overall quality of the segmented image. There are a large number of segmentation quality measures that have been suggested,[2] although none have achieved widespread acceptance as a universal measure of segmentation quality. In order to overcome the drawbacks of using only a single quality measure, we have incorporated an evaluation technique which uses the weighted sum of the five different quality measures as the overall fitness for a particular parameter set. The reward that is generated from this approach is a scalar measurement of the parameter set's utility. However, a more complex vector evaluation which provides multidimensional feedback on segmentation quality can be used.[20] In our initial experiments, we use only a scalar measure of quality for simplicity.

The measures of segmentation quality that we have selected for this work include (weighting shown in parentheses):

(1) *Edge-Border Coincidence* (.20): Measures the overlap of the region borders in the image acquired from the segmentation algorithm relative to the borders found using an edge operator. In our application, we used the Sobel operator to compute the necessary edge information.

(2) *Boundary Discrepancy* (.40): Measures the total number of overlapping boundary pixels in the ground truth image and the segmented image, minus the total number of non-overlapping pixels in these two images.

(3) *Pixel Misclassification* (.15): Measures the number of target pixels misclassified as background pixels and the number of background pixels misclassified as target pixels.

(4) *Target Contrast* (.15): Measures the contrast between the target and the background in the segmented image, relative to the target contrast in the ground truth image.

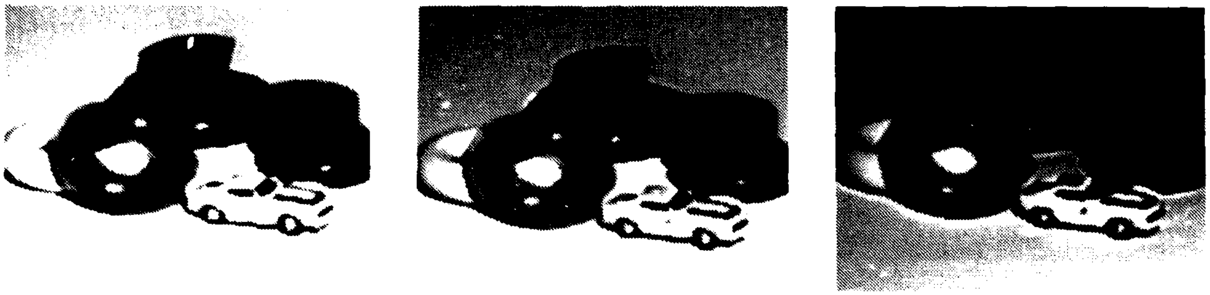
(5) *Target Overlap* (.10): Measures the area of intersection between the target region in the ground truth image and the segmented image, divided by the union of the target regions.

The last four quality measures require the availability of ground truth information which represents the *ideal* segmentation of the image. For these experiments, the ground truth information is acquired by interactively running the PHOENIX algorithm on the image to obtain the best overall results. The weighted sum of the five quality measures is computed once each of the individual measures is known. The fitness of the parameter set is represented by this weighted sum value. In actual applications, the segmentation quality would be provided by a higher level process that was performing region labeling or object recognition and would be able to relate the quality of the segmentation with the region labeling results.

5. INITIAL EXPERIMENTS

The experiments we have performed to date use simplified scenes in which the position and intensity of the lighting have been changed. The variations in each scene are meant to approximate lighting conditions in outdoor imagery, where the angle of the sun varies over time and the intensity changes due to environmental conditions. We have selected three images in order to evaluate the parameter optimization capabilities of the genetic algorithm in the image segmentation domain. The images, shown in Figure 4, are color images (red, green, and blue components) which are 128 by 128 pixels in size. In these initial images, only the lighting intensity has been modified between scenes. The position of the lighting and the camera remain fixed. The car shown at the bottom of images is considered the object of interest for these experiments. The ground truth data for these images was obtained by interactively running the PHOENIX algorithm on Frame 1 (highest contrast image). The ground truth image is shown in Figure 5.

The first issue of concern for the experiments was the selection of the appropriate control parameters for the genetic algorithm itself, e.g. the population size, crossover rate, mutation rate, and maximum number of allowable



Frame 1 Frame 2 Frame 3

Figure 4: Imagery used to evaluate the adaptive image segmentation process. In each image, the lighting intensity has been changed to simulate varying environmental conditions.

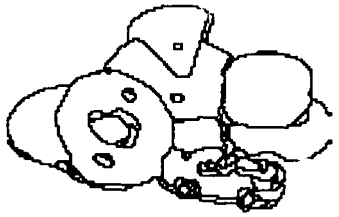


Figure 5: Ground truth data used for evaluating the images shown in Figure 4.

generations (or maximum number of segmentation cycles) of the genetic process. To properly select these values, and further, to validate the performance of genetic process for our image segmentation problem, we exhaustively defined the objective function for the first image (Frame 1). The two segmentation parameters that were selected for adaptation (maxmin and hsmooth) were constrained to a useful range of values and then allowed to take on 32 distinct values within those ranges. Maxmin values, which affect the segmentation quality in a non-linear fashion, were sampled exponentially over a range of values from 100 to 1100. Values near 100 were spaced closer together than values at the upper end of the range. Hsmooth values were sampled linearly, using odd numbers between 1 and 63, inclusive. This allocation provided a search space of 1024 different parameter combinations. For each parameter set, Frame 1 (Figure 4) was segmented and the results were evaluated using the quality measures described in section 4.4. The individual surfaces, along with the combined segmentation quality measure, are shown in Figure 6. Notice that the individual surfaces as well as the combined surface are very complex and can not be effectively optimized using traditional optimization techniques.

Once the surface was exhaustively defined, we were able to apply the genetic algorithm to this search space and efficiently test various combinations of genetic parameters without incurring the cost of image segmentation at every step. Additionally, since we know the maximum value of the surface, we can determine when the adaptive system has created individuals within a given threshold of the maximum. This information provides us with the stopping criterion for the first image. After 30 trial runs using various combinations of genetic parameters, we found that a population size of 10, a crossover rate of 0.8, and a mutation rate of 0.01 produced the fastest convergence rate. This set of genetic parameters allowed the adaptive system, which started from a random selection of individuals, to locate a parameter set that was within 1% of the global maximum value in only three generations (26 total segmentations). Figure 7(a) illustrates the ten randomly generated parameter locations (blackened squares) for Frame 1 at the beginning of the genetic process. Figure 7(b) displays the ten final population locations at the end of the third generation. Note that in Figures 7(a) and 7(b), some population members are not visible due to the viewing angle of the surface. In order to judge the progress of the segmentation quality at each generation, Figure 8 shows the segmentation results for the best (maximum fitness) parameter set at the end of each generation. The sequence show the increase in segmentation accuracy, as compared to the ground truth image in Figure 5, at the end of each generation. By the end of the third generation, the maximum fitness is high enough to halt any further segmentation. The ten members of the population at the end of the third generation are stored for later use in processing images with similar characteristics.

The knowledge acquired in processing Frame 1 was used during the processing of the Frame 2 and Frame 3 shown in Figure 4. Using the same population size, crossover rate, and mutation rate, the images were processed. However, in these two images, the objective function was not exhaustively computed so each individual in the population of each generation resulted in a segmentation cycle followed by a subsequent performance evaluation step. Further, since the maximum value of the objective function was not known, the stopping criteria was selected by stopping after a given number of generations, as determined from evaluating the convergence rate of Frame 1. Because of variations in the shape and maximum value of the objective function, which are caused by the minor differences in all three images in Figure 4, the number of generations before stopping was increased from three to five for Frames 2 and 3 to insure that the final results would be of high quality. Figures 9 and 10 display the best segmentation results obtained at the end of each generation for Frames 2 and 3, respectively.

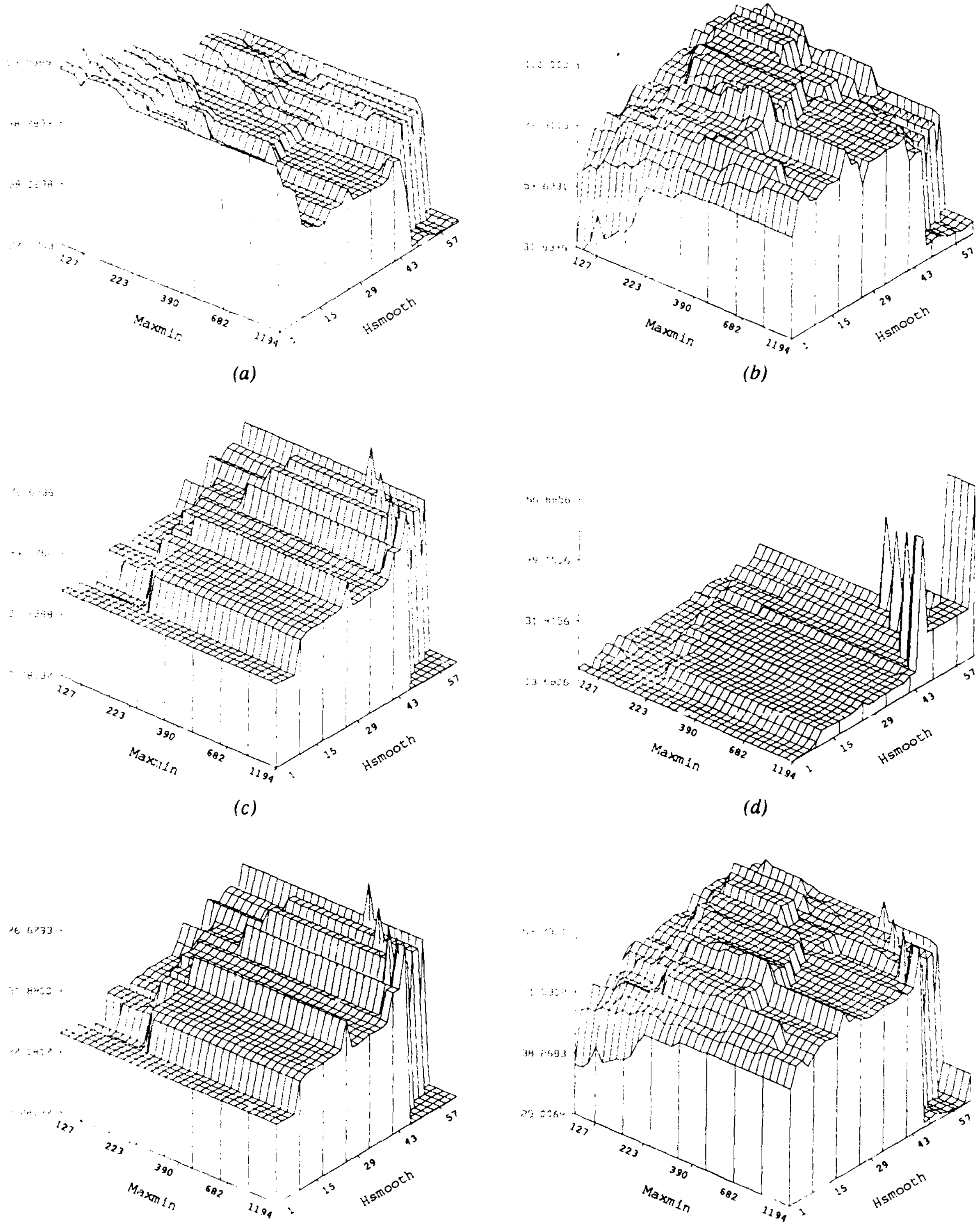
In order to summarize the performance of the genetic algorithm on the objective functions for the three images, Figure 11 charts the maximum and average segmentation performance through 12 complete generations. The adaptive process was allowed to run 100 segmentation iterations to analyze the convergence rate in each of the three images. As the maximum performance chart (Figure 11(a)) shows, the genetic optimization technique achieved maximum results within 5 generations in all cases. In addition, the average performance chart (Figure 11(b)) indicates an upward trend in the performance level of the population members.

6. CONCLUSIONS

We have shown the ability of genetic algorithms to provide high quality segmentation results in a minimal number of segmentation cycles. The knowledge gained during the processing of an image can be stored for later use in a large population of classifiers which can suggest high quality classifiers for images with similar characteristics, thus avoiding the use of random parameter setting during the initial genetic processing. In some cases, the parameter settings suggested by the initial classifiers may produce acceptable segmentation results after only one segmentation cycle.

The next series of experiments currently planned will utilize the final populations created from each of the three images as a larger population from which the genetic learning system can select. As each image is sequentially processed, the new classifiers which are created will replace the weakest members of the current population and the diversity of the classifiers will begin to increase. In this manner, the system will be able to learn from experience and apply this knowledge in succeeding image processing stages.

In addition to implementing the complete genetic learning system just described, we will investigate the adaptation of additional parameter values and measure the improvement in segmentation performance by doing so. There may exist a maximum number of useful parameters, beyond which the cost of adaptation exceeds the overall



(a)

(b)

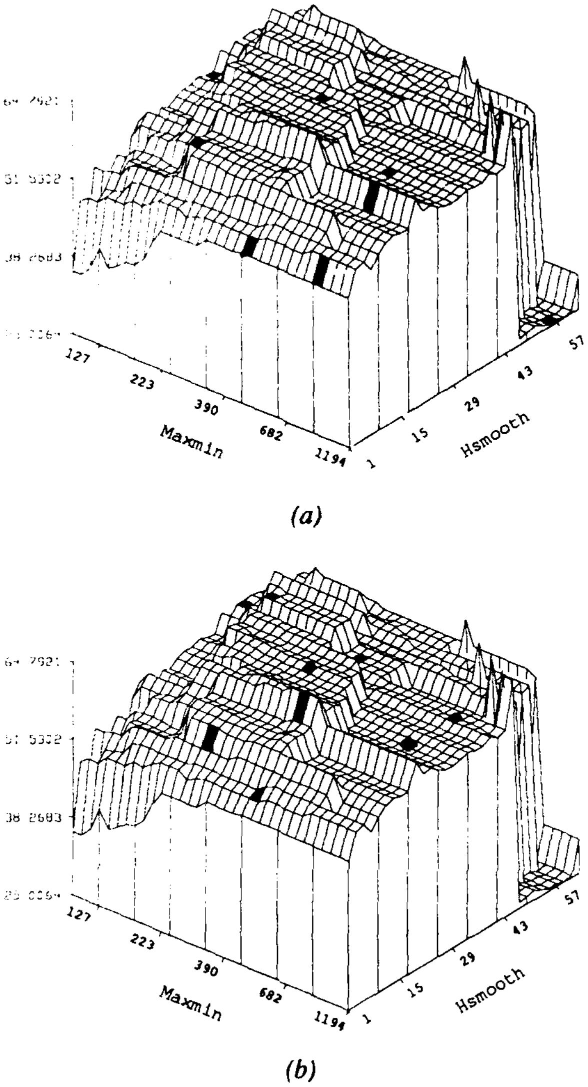
(c)

(d)

(e)

(f)

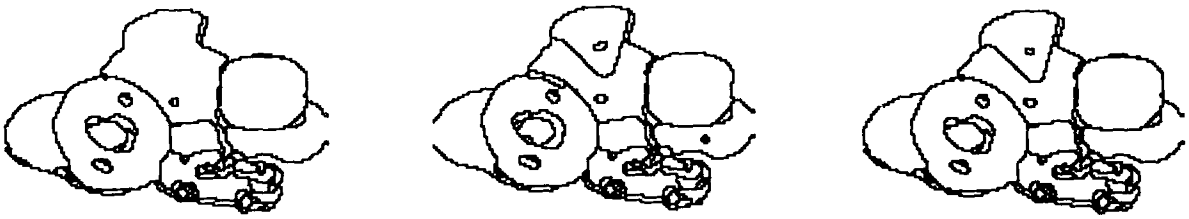
Figure 6: Segmentation quality results obtained for Frame 1 in Figure 4 using various parameter combinations. *(a)* Edge-Border Coincidence. *(b)* Boundary Discrepancy. *(c)* Pixel Misclassification. *(d)* Target Contrast. *(e)* Target Overlap. *(f)* Combined segmentation quality.



(a)

(b)

Figure 7: Search points used during genetic adaptation on Frame 1, indicated as darkened squares on the objective function. *(a)* Randomly selected starting point locations. *(b)* Locations of population members at the end of the third generation, which show an increase in overall population fitness.

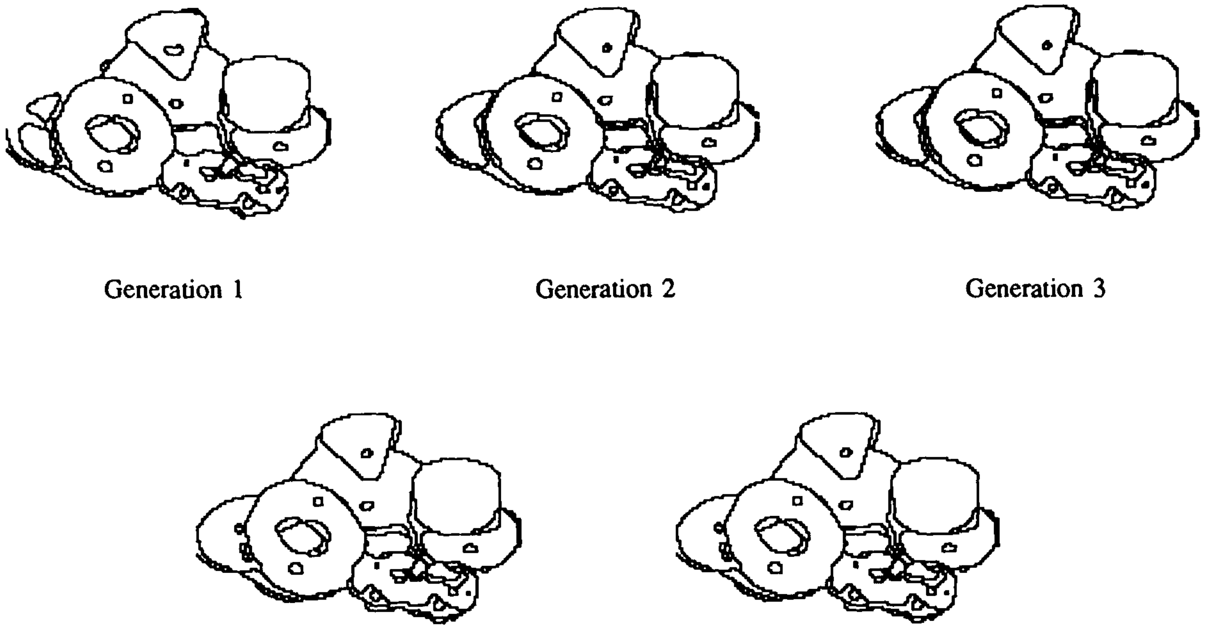


Generation 1

Generation 2

Generation 3

Figure 8: Best segmentation results of Frame 1 at the end of each generation.



Generation 1

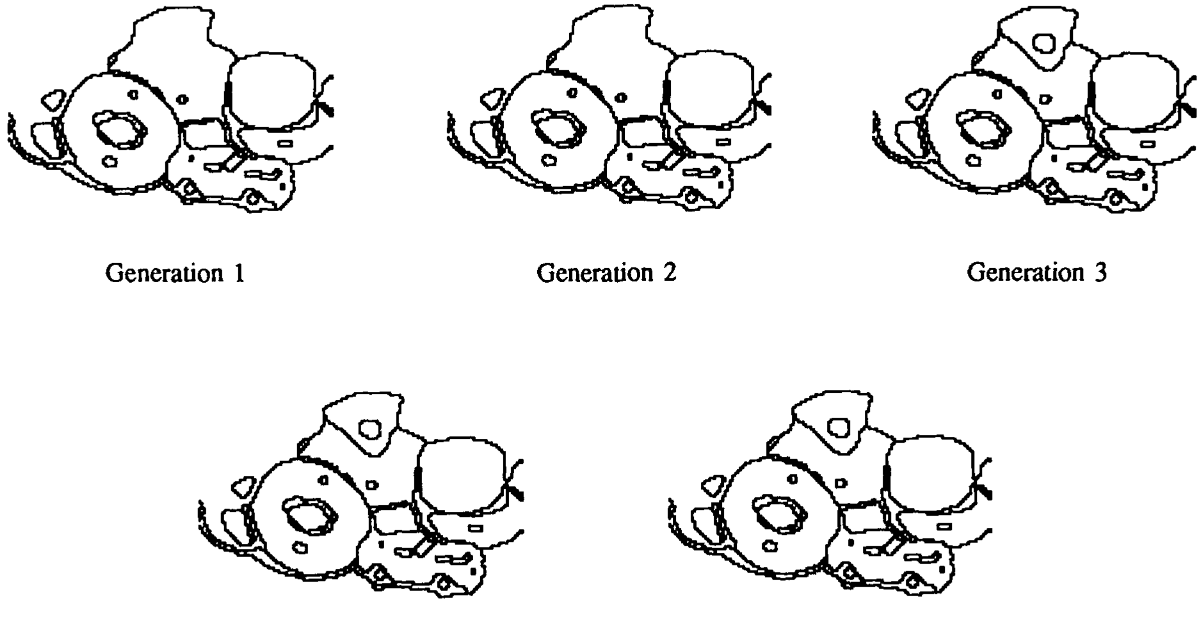
Generation 2

Generation 3

Generation 4

Generation 5

Figure 9: Best segmentation results of Frame 2 (Figure 4) at the end of each generation.



Generation 1

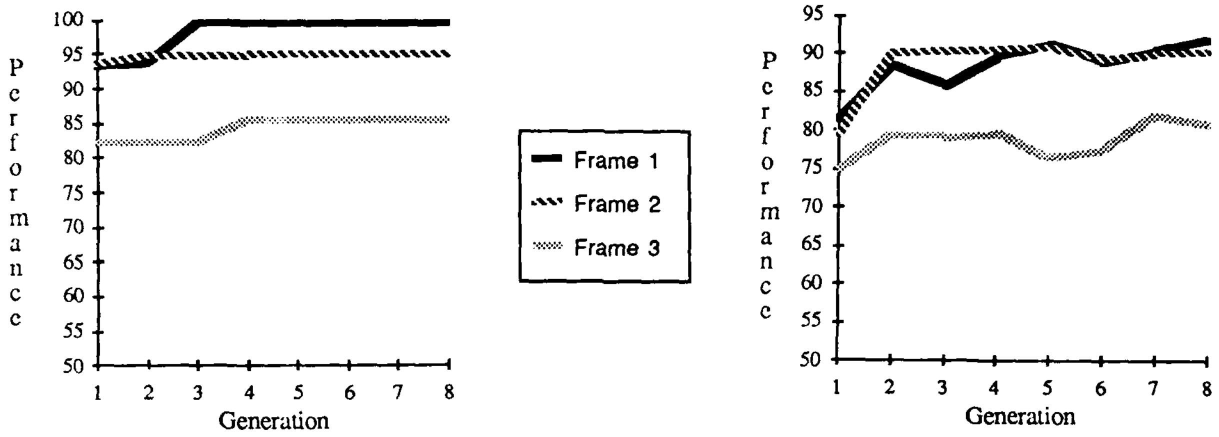
Generation 2

Generation 3

Generation 4

Generation 5

Figure 10: Best segmentation results of Frame 3 (Figure 4) at the end of each generation.



(a) *(b)*

Figure 11: Performance summary for the three images. *(a)* Maximum segmentation performance at the end of each generation. *(b)* Average segmentation performance at the end of each generation.

improvement in segmentation quality. Further research is necessary to evaluate this hypothesis. We also plan to test the adaptive segmentation process on outdoor imagery, in which realistic lighting and other environmental changes can be observed. Using this imagery, we will evaluate the improvement in performance (effectiveness and efficiency) of our adaptive *system versus one with no adaptive capabilities.*

Several important findings are worth mentioning at this point. First, one crucial advantage is that the genetic algorithm can be easily applied to *any* segmentation technique which can be controlled through parameter changes. In addition, we can choose to adapt the entire parameter set or just a few of the critical parameters, depending on the final quality of the results that are desired. Second, it is useful to note that the adaptive segmentation system is only as robust as the segmentation technique which is employed. It cannot cause an algorithm to modify the manner in which it performs the segmentation task. It can only optimize the manner in which the algorithm converges to its best solution for a particular image. However, it may be possible to keep multiple segmentation algorithms available and let the genetic process itself dynamically select the appropriate algorithm based on image characteristics. Another important point is that, although we have only used color images in these experiments, the technique itself is applicable to any type of imagery whose characteristics can properly be represented. This set includes FLIR, LADAR, MMW, and gray scale imagery. Finally, the genetic process described in this paper may soon be able to benefit from advances in parallel computing and VLSI technology, which are now beginning to produce chips that can perform the image segmentation process in real time.[17]

REFERENCES

1. D.H. Ackley, "Stochastic Iterated Genetic Hillclimbing," Ph.D. Thesis, Dept. of Computer Science, Carnegie Mellon University (March, 1987).

2. B. Bhanu, "Automatic Target Recognition: State of the Art Survey," *IEEE Transactions on Aerospace and Electronic Systems* AES-22(4) pp. 364-379 (July, 1986).

3. L. Davis and M. Steenstrup, "Genetic Algorithms and Simulated Annealing: An Overview," in *Genetic Algorithms and Simulated Annealing*, ed. L. Davis, (1987).

4. K. DeJong, "Learning with Genetic Algorithms: An Overview," *Machine Learning* 3 pp. 121-138 (1988).

5. J.M. Fitzpatrick and J.J. Grefenstette, "Genetic Algorithms in Noisy Environments," *Machine Learning* 3 pp. 101-120 (1988).

6. J.M. Fitzpatrick, J.J. Grefenstette, and D. Van Gucht, "Image Registration by Genetic Search," Proceeding of IEEE Southeastern Conference, pp. 460-464 (1984).

7. M.P. Fourman, "Compaction of Symbolic Layout Using Genetic Algorithms," Proceedings of International Conference on Genetic Algorithms and Their Applications, pp. 141-153 (July, 1985).

8. K.S. Fu and J.K. Mui, "A Survey on Image Segmentation," *Pattern Recognition* **13** pp. 3-16 (1981).

9. S. Geman and D. Geman, "Stochastic Relaxation, Gibbs Distributions, and the Bayesian Restoration of Images," *IEEE Transactions on Pattern Analysis and Machine Intelligence* **PAMI-6** pp. 721-741 (1984).

10. A.M. Gillies, "Machine Learning Procedures for Generating Image Domain Feature Detectors," Ph.D. Thesis, Dept. of Computer and Communication Sciences, University of Michigan (April, 1985).

11. D.E. Goldberg, "Computer-Aided Gas Pipeline Operation Using Genetic Algorithms and Rule Learning," Ph.D. Thesis, Dept. of Civil Engineering, University of Michigan (1983).

12. D.E. Goldberg, "Dynamic System Control Using Rule Learning and Genetic Algorithms," Proceedings of International Joint Conference on Artificial Intelligence, pp. 588-592 (1985).

13. J.J. Grefenstette, R. Gopal, B.J. Rosmaita, and D. Van Gucht, "Genetic Algorithms for the Traveling Salesman Problem," Proceedings of International Conference on Genetic Algorithms and Their Applications, pp. 160-168 (July, 1985).

14. R.M. Haralick and L.G. Shapiro, "Image Segmentation Techniques," *Computer Vision, Graphics, and Image Processing* **29** pp. 100-132 (1985).

15. J.H. Holland, *Adaptation in Natural and Artificial Systems*, Univeristy of Michigan Press (1975).

16. J.H. Holland, "Escaping Brittlenes: The Possibilities of General-Purpose Learning Algorithms Applied to Parallel Rule-Based Systems," pp. 593-623 in *Machine Learning: An Artificial Intelligence Approach, Vol. II*, ed. R.S. Michalski, J.G. Carbonell, and T.M. Mitchell,Morgan Kaufmann Publishers, Inc. (1986).

17. B.L. Hutching, B. Bhanu, and K.F. Smith, "Computer Aided VLSI Design and Implementation of a Real-Time Segmentation Processor," *Submitted to Machine Vision and Applications International Journal*, (1989).

18. K.I. Laws, "The Phoenix Image Segmentation System: Description and Evaluation," SRI International Technical Note No. 289 (December, 1982).

19. R. Ohlander, K. Price, and D.R. Reddy, "Picture Segmentation Using a Recursive Region Splitting Method," *Computer Graphics and Image Processing* **8** pp. 313-333 (1978).

20. J.D. Schaffer, "Multiple Objective Optimization with Vector Evaluated Genetic Algorithms," Proceedings of International Conference on Genetic Algorithms and Their Applications, pp. 93-100 (1985).

21. S. Shafer and T. Kanade, "Recursive Region Segmentation by Analysis of Histograms," Proceedings of IEEE International Conference on Acoustics, Speech, and Signal Processing, pp. 1166-1171 (1982).

22. D. Smith, "Bin Packing with Adaptive Search," Proceedings of International Conference on Genetic Algorithms and Their Applications, pp. 202-206 (July, 1985).

23. S.F. Smith, "Flexible Learning of Problem Solving Heuristics Through Adaptive Search," Proceedings of 8th International Joint Conference on Artificial Intelligence, pp. 422-425 (1983).

24. E.D. Sontag and H.J. Sussmann, "Image Restoration and Segmentation using the Annealing Algorithm," Proceedings of 24th Conference on Decision and Control, pp. 768-773 (December, 1985).

IMAGE AND BOUNDARY SEGMENTATION VIA MINIMAL-LENGTH ENCODING ON THE CONNECTION MACHINE[1]

Yvan G. Leclerc

Artificial Intelligence Center
SRI International
333 Ravenswood Avenue
Menlo Park, California 94025

Abstract

We present a hierarchical optimization approach to the image partitioning problem: that of finding a complete and stable description of an image, in terms of a specified descriptive language, that is simplest in the sense of shortest description length. The first stage in the hierarchy uses a low-order polynomial description of the intensity variation within each region and a chain-code-like description of the region boundaries. By using a regular-grid finite-element representation for the image, the optimization technique, called a *continuation method,* reduces to a simple, local, parallel, and iterative algorithm that is ideally suited to the Connection Machine^{TM}. Also presented are some preliminary results of the second stage, in which the boundaries of the regions are described in terms of straight-line segments. In this case, a parallel implementation of a dynamic-programming algorithm is used to find the simplest description.

Introduction

Vision can be thought of as an inference process—one in which a description of the outside world is inferred from images of the world, prior information about the world, and prior information about the image sensor. The particular kind of inference process used in this paper might be called "inference to the simplest explanation" or, more formally, "minimal-length encoding" [1,8,11,21,23,25,28,29]. The basic idea is that prior information about the world and the sensor is incorporated in the language used to describe the world and sensor, and the inference process is to find the simplest (i.e., shortest) description in that language that exactly reproduces the given images.

The basic motivation behind this inference process is the following hypothesis. If one can find a language that provides an efficient description of a large number of observations (images), then the simplest descriptions in that language tell us something about the causes of the observations.

For example, consider a sequence of images of a static scene. Certainly, a complete three-dimensional description of the shape and color of objects and light sources, plus a description of the camera parameters would be extremely efficient because, having once described the scene (however complex that may be), each image can be reproduced by simply specifying the camera parameters and invoking a rendering algorithm. Moreover, one can argue that, for sufficiently complex scenes and for a sufficiently large number of distinct images, no other language could ever be as efficient without also decomposing the description into three-dimensional objects and a camera. Thus, in this example at least, simplicity of description leads us to the correct decomposition of the causes of the images.

When viewed from this perspective, the solution to the computer vision problem has two parts. First is the design of an efficient language for describing images based on our understanding of the causal processes that combine to form images. Second is the design of computationally effective procedures for finding the simplest descriptions in this language. However, because of the enormous search spaces, these two aspects cannot be so neatly separated—the language must undoubtedly be designed in such a way as to facilitate the search for simple descriptions.

One way of combining these two aspects is to design a hierarchical descriptive language such that incrementally more efficient descriptions can be obtained at each level via an incremental decomposition of the image into ever smaller groups of causal processes. At the first level (the image), all of the causal processes are grouped into a single description—an array of intensities.

[1] The work reported here was partially supported by the Defense Advanced Research Projects Agency under contract MDA903-86-C-0084.

I propose that at the second level, the image be decomposed into two groups of causal processes: the projection of an idealized world onto an idealized image plane, and the deviations from this projection due to small-scale texturing of objects, sensor noise, and the point-spread function of the image sensor. The complete language for describing this decomposition, and the complete algorithm for finding the simplest description in this language, are described in another paper by this author [15]. The algorithm is a significant improvement over that first presented in [16]. In this paper, we present the language and a Connection Machine^{TM} (CM) implementation of the algorithm for the special case in which the image is decomposed into the sum of an underlying piecewise-constant image and white noise with known variance. This underlying image is described by regions of constant intensity, and the boundaries of the regions are described by a chain-code boundary.[2]

Having thus removed the stochastic deviations at the second level, it should be possible to decompose the description of the underlying image into two or more groups of causal processes at the third level. A first step in this direction is presented here by finding the simplest description of the chain-coded region boundaries in terms of straight lines using a parallel dynamic programming algorithm implemented on the CM. The parallelism reduces the $O(n^2)$ dynamic programming algorithm to an $O(n)$ algorithm, where n is the number of elements in the chain-code.

Continuing in this manner, one should eventually decompose the description of the image into its individual causal processes and their interrelationships. How such a hierarchy might differ from more traditional computer vision ones, such as Barrow and Tenenbaum's "Intrinsic Images" [2], Marr's "$2\frac{1}{2}$-D Sketch" [19], or those emerging from neurophysiology and psychophysics [7,13], remains to be seen.

In the next section, we present the mathematics and implementation of the special case of a piecewise-constant underlying image. In short, the problem of finding the shortest description is posed as a global optimization problem, wherein the objective function is directly related to the description length. Because of the nonlinearities induced by the discontinuous nature of the underlying image, the objective function is highly nonconvex, so that standard optimization techniques cannot find the global minimum. Instead, a technique called a *continuation method* is used. This technique uses a regular-grid, finite-element representation for the underlying image. With this representation, the continuation method reduces to a simple, local, parallel, and iterative algorithm that is ideally suited to such massively parallel architectures as the CM. Results and timings of the implementation are presented.

Following this, we present the details of the language and algorithm for describing the chain-coded region boundaries in terms of straight lines using a parallel dynamic programming algorithm implemented on the CM. Results and timings of the implementation are presented.

Image Segmentation

For this paper, we shall only consider in detail the special case in which a real image is the sum of an underlying piecewise-constant image and white noise with known variance. The more general case of an underlying piecewise-smooth image and white noise with unknown variance is treated elsewhere [15]. See also the work of Pednault [23] for a dynamic-programming solution to finding the simplest description for one-dimensional signals.

We denote the real $n \times m$ image by the vector $\mathbf{z}$ indexed by $i \in I = 1, \ldots, nm$. The underlying image $u(x,y)$ is represented by a regular grid of square 1×1 elements, with each element centered at the coordinate (x_i, y_i) of the i^{th} pixel in the real image. The 1×1 square centered at (x_i, y_i) is the *spatial domain* $\mathcal{X}_i$ of the i^{th} element, and the value of the element is u_i. Thus,

$$u(x,y) = u_i \qquad \forall\, (x,y) \in \mathcal{X}_i,\ i \in I,$$

and the underlying image is completely represented by the vector $\mathbf{u} = \{u_i,\ i \in I\}$.

Similarly, we represent the noise by the vector $\mathbf{r}$. Thus, the statement that the real image is the sum of the underlying image and the noise can be written as

$$\mathbf{z} = \mathbf{u} + \mathbf{r}. \tag{1}$$

A consequence of this choice of representations is that discontinuities in the underlying image can occur only along the vertical and horizontal boundaries between the grid elements. One advantage of this is that the underlying image is uniquely specified when there is no noise (namely, $\mathbf{u} = \mathbf{z}$). However, a more sophisticated representation in which elements have variable shape is also possible. This is an excellent avenue for future research.

Using the above definitions, the problem of finding the simplest description is therefore

$$(\mathbf{u}^*, \mathbf{r}^*) = \min_{(\mathbf{u},\mathbf{r}): \mathbf{z}=\mathbf{u}+\mathbf{r}} |\mathcal{L}_u(\mathbf{u})| + |\mathcal{L}_r(\mathbf{r})|,$$

[2] Although only this special case is presented here, the complete algorithm has been implemented on the CM.

where $\mathcal{L}_u$ and $\mathcal{L}_r$ denote the languages used to describe $\mathbf{u}$ and $\mathbf{r}$. From Eq. 1, the equivalent problem is

$$\mathbf{u}^* = \min_{\mathbf{u}} |\mathcal{L}_u(\mathbf{u})| + |\mathcal{L}_r(\mathbf{z} - \mathbf{u})|.$$

There are two steps involved in solving this problem. First, we must define the languages $\mathcal{L}_u$ and $\mathcal{L}_r$. Second, we must specify a computationally feasible procedure for finding $\mathbf{u}^*$ and for determining the stability of the solution.

Defining Descriptive Languages

The first task, then, is to define a language for describing the underlying piecewise-constant image $\mathbf{u}$. By definition, $\mathbf{u}$ is composed of regions of constant intensity. Thus, for each region, we need specify only the shape and position of the region boundaries and the constant intensity within the region. The region boundaries are described by a chain code of unit-length line segments located between adjacent elements; each line segment corresponds to the boundary between adjacent square grid elements. The number of bits required to describe each region is thus proportional to the number of elements in the chain plus a constant to specify the constant intensity and the first element of the chain. The total number of bits required to specify the underlying image is thus proportional to the number of regions plus the total length of the region boundaries.

Because region boundaries occur only when spatially adjacent elements of $\mathbf{u}$ are different, their total length can be determined locally by counting all adjacent elements (u_i, u_j) that have a nonzero difference and dividing by 2 (because region boundaries will be counted twice this way). Thus, the total length of the region boundaries is

$$\frac{1}{2}\sum_{i\in I}\sum_{j\in N_i}(1-\delta(u_i-u_j)),$$

where

$$\begin{aligned} N_i &= \text{the set of neighbors of the } i^{th} \text{ element} \\ \delta(x) &= \text{the Kronecker delta} = \begin{cases} 1 & \text{if } x = 0 \\ 0 & \text{otherwise} \end{cases} . \end{aligned}$$

When the regions are relatively large, a good approximation to the number of bits required to describe $\mathbf{u}$ is thus

$$|\mathcal{L}_u(\mathbf{u})| \approx \frac{b}{2}\sum_{i\in I}\sum_{j\in N_i}(1-\delta(u_i-u_j)), \tag{2}$$

where b is the sum of (1) the number of bits required to encode each element in the chain code and (2) the number of bits required to encode the constant intensity and starting element, divided by the average region-boundary length.

As for describing the noise, the fewest bits required to describe data generated by a stochastic process is the negative base-two logarithm of the probability of observing that data [25]. Because we assume the noise to be uncorrelated,

$$\begin{aligned} |\mathcal{L}_r(\mathbf{r})| \equiv -\log_2 P(\mathbf{r}) &= -\log_2 \prod_{i\in I} P(r_i) \\ &= -\sum_{i\in I} \log_2 P(r_i). \end{aligned}$$

Furthermore, we assume the noise to be quantized white noise, where the elements are drawn from a normal distribution and then quantized to the nearest q, the precision of the pixels in the real image. Thus,

$$\begin{aligned} P(r_i) &= \int_{\lfloor r_i \rfloor_q}^{\lceil r_i \rceil_q} \frac{1}{\sqrt{2\pi\sigma^2}} \exp\left(\frac{-x^2}{2\sigma^2}\right) dx \\ &\approx \frac{q}{\sqrt{2\pi\sigma^2}} \exp\left(\frac{-r_i^2}{2\sigma^2}\right) \quad \text{when } q < \sigma, \end{aligned} \tag{3}$$

and

$$-\log_2 P(\mathbf{r}) \approx nm\,c + a\sum_{i\in I}\left(\frac{r_i}{\sigma}\right)^2. \tag{4}$$

Thus, for $\mathbf{u}$ and $\mathbf{r}$ satisfying Eq. 1, an approximation to the total number of bits required to describe $\mathbf{u}$ and $\mathbf{r}$ is

$$|\mathcal{L}_u(\mathbf{u})| + |\mathcal{L}_r(\mathbf{r})| \approx nm\,c + L(\mathbf{u}),$$

where

$$L(\mathbf{u}) = a \sum_{i \in I} \left(\frac{z_i - u_i}{\sigma} \right)^2 + \frac{b}{2} \sum_{i \in I} \sum_{j \in N_i} (1 - \delta(u_i - u_j)) . \tag{5}$$

Dropping the additive constant, the minimization problem can thus be written as

$$\mathbf{u}^* = \min_{\mathbf{u}} L(\mathbf{u}).$$

Defining a Computationally Feasible Procedure

The simplest, direct way of finding the global minimum of $L(R)$ is to search through all possible sets of regions, calculating the cost for each set, and choosing the set with the smallest cost. Unfortunately, the number of possible sets of regions grows exponentially with the number of elements of $\mathbf{u}$, rendering such a search completely infeasible. Even dynamic programming-like algorithms require at least the evaluation of the cost for every possible simple region, which is an exponential in nm when n and m are greater than 1, again rendering such a search computationally infeasible.

Furthermore, because of the Kronecker delta term, $L(\mathbf{u})$ has many local minima. Thus, standard descent-based optimization techniques are useless. Also, the simulated-annealing style of algorithms exemplified in Geman and Geman [10] are inappropriate, because the time complexity is much too high for this type of function [4]. Intuitively, the reason that stochastic gradient-descent algorithms are inappropriate for this particular objective function is that the function has extremely narrow (in fact, infinitesimally narrow) valleys, so that even stochastic sampling of the surface provides no guidance for the search.

Instead, I have devised an algorithm that yields something very close or equal to the optimal solution for a large class of inputs. It belongs to a class of optimization techniques generally called continuation methods [9,32]. This algorithm is similar in spirit to the algorithm described in Blake and Zisserman [5] as the "graduated nonconvexity," or GNC algorithm.

As used here, a continuation method embeds the objective function in a family of functions $L(\mathbf{u}, s)$ for which there is a single local minimum at some large s, and for which the number and position of the local minima converge to those of $L(\mathbf{u})$ as s approaches zero. The steps of the continuation method are straightforward. First, find the unique local minimum $\mathbf{u}^0$ of $L(\mathbf{u}, s^0)$ for some sufficiently large s^0. Then, track the local minimum in $\mathbf{u}$ as a decreasing function of s, as follows. For $s^{t+1} = s^t$, let $\mathbf{u}^{t+1}$ be the result of taking a single step of a descent algorithm, as applied to the objective function $L(\mathbf{u}, s^{t+1})$ started at $\mathbf{u} = \mathbf{u}^t$. When the descent algorithm converges, let $s^{t+1} = r\,s^t$ for some $0 < r < 1$, and repeat until s^t is sufficiently small. For an ideal embedding, there will be no bifurcations along this path, and the value of $\mathbf{u}^t$ for a sufficiently large t (and hence a sufficiently small s^t) will be close or equal to the global minimum of $L(\mathbf{u})$.

The specific embedding used here replaces $\delta(u_i - u_j)$ with an exponential,

$$\delta(u_i - u_j) \rightarrow e_{i,j}(\mathbf{u}, s) \equiv \exp\left(-\frac{(u_i - u_j)^2}{(s\sigma)^2} \right),$$

so that

$$L(\mathbf{u}, s) = a \sum_{i \in I} \left(\frac{z_i - u_i}{\sigma} \right)^2 + \frac{b}{2} \sum_{i \in I} \sum_{j \in N_i} (1 - e_{i,j}(\mathbf{u}, s)) . \tag{6}$$

This is an appropriate embedding because

$$\lim_{s \to 0} e_{i,j}(\mathbf{u}, s) = \delta(u_i - u_j)$$

so that

$$\lim_{s \to 0} L(\mathbf{u}, s) = L(\mathbf{u}),$$

and, hence, the local minima of $L(\mathbf{u}, s)$ approach the local minima of $L(\mathbf{u})$. Furthermore, there exists a unique local minimum of $L(\mathbf{u}, s)$ for sufficiently large s, namely $\mathbf{u} = \mathbf{z}$. This is so because (1) $L(\mathbf{u}, s) \geq 0 \;\forall \mathbf{u}$, (2) $\mathbf{u} = \mathbf{z}$ is the unique point for which the first summation of Eq. 6 is identically zero, and (3) the second summation vanishes for arbitrarily large s when $\mathbf{u}$ is bounded. Thus, for s approaching infinity, $\mathbf{u} = \mathbf{z}$ is the unique point for which $L(\mathbf{u}, s) = 0$, the unique local (and global) minimum.

Intuitively, the exponential term introduces broad valleys when s is large, and converges to the narrow valleys in the limit as s goes to zero. Thus, the continuation method creates a kind of "scale space" representation of the objective function $L(\mathbf{u})$ (in analogy to Witkin's scale-space representation of a signal [31]) and tracks a local minimum from the coarsest scale (where there is only one local minimum) to the finest scale (where there are many).

Although any iterative descent algorithm can be used for the continuation method (see, for example, the wide variety described in Luenberger's excellent book [18]), the following algorithm has proven to be quite efficient for the objective function examined here. Some experimentation with a conjugate-gradient algorithm has, so far, reduced the number of iterations by only a factor of two, but each step of the algorithm is about twice as long as the simpler one below.

By definition, local minima of $L(\mathbf{u}, s)$ occur when

$$\frac{\partial L(\mathbf{u}, s)}{\partial u_i} = \frac{2a}{\sigma^2}(u_i - z_i) + \frac{2b}{(s\sigma)^2} \sum_{j \in N_i} e_{i,j}(\mathbf{u}, s)(u_i - u_j) = 0, \tag{7}$$

which can be written in vector notation as:

$$\nabla L(\mathbf{u}, s) \equiv \frac{\partial L(\mathbf{u}, s)}{\partial \mathbf{u}} = \mathbf{b} + \mathbf{A}(\mathbf{u}, s)\mathbf{u} = 0, \tag{8}$$

where

$$\begin{aligned}
a_{i,i}(\mathbf{u}, s) &= \frac{2a}{\sigma^2} + \frac{2b}{(s\sigma)^2} \sum_{j \in N_i} e_{i,j}(\mathbf{u}, s) \\
a_{i,j}(\mathbf{u}, s) &= \begin{cases} \dfrac{-2b}{(s\sigma)^2} e_{i,j}(\mathbf{u}, s) & \text{if } j \in N_i \\ 0 & \text{otherwise} \end{cases} \\
b_i &= \frac{-2az_i}{\sigma^2}.
\end{aligned}$$

At each step of the iterative descent algorithm, we linearize the above set of equations by setting $s^{t+1} = rs^t$ and fixing $\mathbf{A}^t \equiv \mathbf{A}(\mathbf{u}^t, s^{t+1})$. Because $\mathbf{A}^t$ is diagonally dominant, a Gauss-Seidel iterate can be used to provide a step in the direction of the solution:

$$u_i^{t+1} = \frac{-1}{a_{i,i}^t}\left(b_i + \sum_{j \neq i} a_{i,j}^t u_j^t\right) = \frac{z_i + \dfrac{b}{a(s^{t+1})^2} \displaystyle\sum_{j \in N_i} e_{i,j}^t u_j^t}{1 + \dfrac{b}{a(s^{t+1})^2} \displaystyle\sum_{j \in N_i} e_{i,j}^t}, \tag{9}$$

where

$$e_{i,j}^t \equiv e_{i,j}(\mathbf{u}^t, s^{t+1}).$$

This is carried out on the CM by assigning each element to a virtual processor in a two-dimensional VP set, and iterating in parallel. The *interaction strengths* $e_{i,j}^t$ are recomputed at each iteration via the NEWS network.

The above is repeated until $|u_i^{t+1} - u_i^t|$ is sufficiently small (less than $0.1s^{t+1}\sigma$) for all i; only one or two iterations are typically required to achieve this accuracy. Once convergence has been achieved, s is decreased ($s^{t+1} = rs^t$, $0 < r < 1$), and everything repeated until s^{t+1} is sufficiently close to zero.

When the interaction strength falls below $1/e$ (i.e., when $|u_i^t - u_j^t| < s^{t+1}\sigma$), we say that a [tentative] discontinuity between adjacent elements has been found at time t. The discontinuity is called tentative because it is possible (though relatively rare) for the interaction strength to oscillate a few times before converging to a stable value. The word "tentative" will be dropped unless ambiguity would result. The first value of s^{t+1} for which this occurs is called the *stability*, $s_{i,j}$, of the discontinuity.

The reason for calling $s_{i,j}$ a stability measure, as discussed in detail in another paper [15], is that $s_{i,j}$ is approximately equal to the ratio of the local contrast to σ. Thus, when the contrast is sufficiently large relative to σ, the boundary is typically unaffected by small changes to the input image, whereas when the ratio is low, boundaries can shift unpredictably or disappear altogether. Thus, to obtain a stable description, it is necessary to stop the procedure at a reasonably large value of s^{t+1} (typically 1/4 or so). A different strategy might be to stop at a much smaller value, but then use the stability measure in the subsequent stages.

Results

Because of time and graphics software constraints, the figures in this paper were all produced using a Symbolics 3640 Lisp-Machine. Similar, but not identical, results were obtained using the CM. The primary reason for the difference is that the Lisp-Machine implementation does not update every element at each Gauss-Seidel iteration. To save time, only those elements that differ significantly from the previous iteration are updated (except, of course, all elements are updated at the iteration when s^t is decreased). This results in a significant increase in speed on a sequential machine, but also produces a slight degradation in performance. This was not fully appreciated before the CM implementation.

For a 128 × 128 input image, with a VP-ratio of 2 and without floating-point hardware, the CM takes about 0.7 second per iteration for the piecewise-constant case, which is about 100 times faster than the Lisp-Machine implementation (when all elements are updated at each iteration). For the piecewise-first-order case, the time increases to about 3.8 seconds per iteration. In general, the time is approximately $k^2/2$ times the time required for the piecewise-constant case, where k is the number of coefficients in the highest-order polynomial (three for first-order polynomials, six for second order, and so on).

The results presented here were obtained by using the most general form of the encoding-length function, in which the underlying image is piecewise polynomial, the variance of the noise is unknown and piecewise constant, and the sensor model includes a point-spread function. A key point about these examples is that they were all obtained by using *precisely the same parameters*, with the following exceptions. First, a Gaussian point-spread function with $\sigma = 1$ was used for all of the real images, but no point-spread function was used for any of the synthetic images (taking advantage of our *a priori* knowledge about how these synthetic images were created). Second, for demonstrative purposes only and as noted for each example, several values of p_{max}, the order of the underlying image, were used. The conclusion that emerges from these and many other examples not presented here is that a piecewise-second-order underlying image is appropriate for a large class of real images.

The first example illustrates the power of global optimization compared with purely local, noniterative, operations. Figure 1a is the 20 × 20 input image, which is the sum of a piecewise-first-order image and zero-mean white noise with unit variance. The outer region of the underlying image has intensity 0.0, the center ramp has a slope of 1.0, and the contrast at either end of the ramp with the outer region is 4.0. Of course, the contrast of the center of the ramp with the background is 0.

Figures 1b and 1c illustrate the result of the procedure for $p_{max} = 1$ and 2, respectively, stopping at $s^t = 1/4$. First, note that the entire ramp is separated from the background, even in the center where the local signal-to-noise ratio is 0 (the thinner line separating the ramp from the background near the center indicates that the discontinuity is only of order 1, that is, a discontinuity in the first derivative of the underlying image). This is in contradistinction to the output of the Canny edge detector [6]. For a small spatial scale (Figure 1d), the Canny operator leaves a gap (not to mention the introduction of spurious discontinuities due to the assumption that edges are locally piecewise-constant), whereas a larger spatial scale (Figure 1e) simply makes the artifacts worse. (The operator was unable to find the correct outline for any parameter settings.) Second, note that the elements of the ramp have been determined to be order 1 (as indicated by the number immediately above each element, no number means that the element is order 0), whereas the elements of the outer region have been determined to be order 0. Thus, the procedure has not only located the discontinuities correctly, but has also determined the correct order for each region.

Figure 2 illustrates an application of the procedure to an aerial image of a house, with $p_{max} = 1$, stopping at $s^t = 1/4$. Figures 2b and 2c show the resulting underlying image and discontinuities. Figure 2d is an image of the stability measure for these discontinuities, with the darkest lines indicating the most stable discontinuities. Two interesting points emerge from this example. First, the four bushes in the upper-left corner are almost completely delineated, even though the contrast along that part of their boundaries is virtually nil. This is an example of the "zero contrast" situation similar to the previous synthetic ramp image. Second, the majority of discontinuities that form closed regions have high stability measures. This is a fairly strong indication that the piecewise-first-order (or higher-order) model is appropriate for this image. To verify this conclusion, observe that the discontinuities obtained using $p_{max} = 2$ (Figure 3) are virtually identical, the only exceptions being the few very-low-stability discontinuities.

Figure 4 illustrates an application of the same model with $p_{max} = 1$ (using precisely the same parameters) to the image of a face. In this example, about half the discontinuities have a fairly low stability measure. This indicates that the language is probably not appropriate for this image. This is especially evident in the cheek and chin areas where a higher-order model is clearly more appropriate. Even so, the discontinuities with high stability measures appear to be good candidates for region boundaries. Figure 5 shows the results for $p_{max} = 2$, in which the artifacts due to using too low an order are entirely absent.

Boundary Segmentation

Having decomposed the original image into an underlying piecewise-smooth image and noise, the hierarchical encoding scheme proposed in the introduction demands that we describe the underlying image in increasingly simpler fashions, eventually leading to descriptions in terms of the three-dimensional structure of the scene. (Note that, if we've done the decomposition correctly, the noise cannot be described in any simpler fashion than with the optimal language described above.) The first steps in this direction might be to describe the region boundaries more compactly in terms of straight lines and smooth curves, and to group together non-adjacent regions that can more simply be described jointly than independently.

Some of the reasons that grouping non-adjacent regions can lead to simpler descriptions are (1) the intensities within the regions are more compactly encoded using a single polynomial than several independent ones, (2) the positions of the regions are related (e.g., many small regions might form a smooth curve or a flow field [33]), (3) parts of the region boundaries are related (e.g., the parts might form a straight line or smooth curve, or they might be parallel), and so on. The information required to perform the first two types of grouping are already explicit in the description of the underlying image. The third type of grouping, however, requires a more sophisticated representation of the region boundaries than the simple chain-code used above. This is an additional motivation for describing the region boundaries in terms of straight lines and smooth curves.

As a first step, then, we describe boundaries in terms of straight lines (see the paper by Smith and Wolf [27] for a similar application of minimal-length encoding). In analogous fashion to the image segmentation presented above, each segment of the boundary is described in terms of a straight line and the deviations from the line. Given that the straight lines form a continuous contour, each segment can be described with two parameters (dx_l, dy_l) being the (x_l, y_l) coordinates of the end of the l^{th} line relative to (x_{l-1}, y_{l-1}). According to Rissanen [25], each parameter can be encoded using $\log^* |n| + 1$ bits, where n is dx_l or dy_l, and

$$\log^* |n| = \log_2 |n| + \log_2 \log_2 |n| + \ldots \text{ for all positive terms}$$

is the number of bits required to encode a positive integer n.

As a first approximation to the cost of encoding the deviations, we encode the boundary points (x_i, y_i) as if their perpendicular distances from the line, r_i, were white noise. Thus, the cost of encoding the deviations r_i to within precision q is:

$$\sum_i \frac{1}{\log 2}\left(\frac{1}{2}\log 2\pi + \log \sigma_l - \log q\right) + \frac{1}{2\log 2}\left(\frac{r_i}{\sigma_l}\right)^2 . \qquad (10)$$

(see the derivation of Eq. 4). The parameters of the line that minimize this cost function can be determined using a standard least-squares algorithm.

The variance parameter σ_l for the l^{th} line segment is either an *a priori* estimate of the variance, or can be computed independently for each line as

$$\sigma_l = \sqrt{\frac{1}{\sum_i 1} \sum_i r_i^2},$$

which is the value that minimizes the encoding length in Eq. 10 above. In the latter case, the variance parameter σ_l must also be encoded for each segment. Rissanen [25] suggests that for an optimal precision, the number of bits required to encode a parameter such as this is approximately $\frac{1}{2}\log_2 n$ plus a constant, where n is the length of the segment.

Ideally, we would like to find the simplest description of all of the boundaries, including appropriate decisions about the branch points that occur when two or more boundaries meet at a single point. For this paper, however, we encode the boundaries of each closed region independently. This is done via a dynamic-programming algorithm where the cost of optimally encoding boundary points 0 to i is defined recursively as

$$C_{0,i} = \min_{j<i} C_{0,j-1} + C_{j,i}.$$

Here, $C_{j,i}$ is the cost of encoding points j to i with a single line segment (using the cost estimates defined above). This formulation, of course, assumes that each segment is encoded independently, even though we would like the end-point of one line segment to coincide with the start-point of the next. Furthermore, boundaries forming closed curves must be arbitrarily cut to form a linear sequence of points, perhaps forcing a sub-optimal segmentation. This sub-optimality can be mitigated somewhat by repeating the set of points several times, and using the segmentation for the set of points near the middle.

Without going into all of the details, the basic idea of the parallel CM implementation of this algorithm is to assign a processor to each boundary point, such that processor i has point (x_i, y_i) in its memory. Such quantities as

$$\sum_{k=0}^{i} x_k^2$$

required by the least-squares algorithm, are computed for all i using the SCAN!! operator (this only takes $O(\log p)$ time, where p is the number of processors). For a given value of i (starting with $i = 0$ until $i = n - 1$), the cost $C_{j,i}$ is computed for each processor for which $j < i$. To compute this cost, such quantities as

$$\sum_{k=j}^{i} x_k^2$$

are required. This can be computed in processor j as the difference of two previously computed sums storeu in processors i and j:

$$\sum_{k=j}^{i} x_k^2 = x_j^2 + \left(\sum_{k=0}^{i} x_k^2 - \sum_{k=0}^{j} x_k^2 \right).$$

In this way, the least-squares parameters and the ensuing encoding cost can be computed in constant time for each processor. Once $C_{j,i}$ has been computed in each processor, the SCAN!! operator can again be used to find the minimum of the sum $C_{0,j-1} + C_{j,i}$, which is then stored in processor i (along with the processor number that contained the minimum sum). The total time required to find the minimal-encoding is thus $O(n)$.

The above algorithm can in fact be applied to all of the boundaries in parallel by placing the boundary points for the first region in the first n_1 processors, then the points for the next region in the next n_2 processors, and so on. By storing a unique region number in each processor (such that all the processors for region 0 have 0 stored in their memory, etc.), all of the SCAN!! operations above can be applied to each boundary independently and in parallel using the so-called segmented-scan feature of the CM. Thus, the total time required to find the minimal-encoding for all of the boundaries is simply proportional to the number of points in the longest boundary.

The result of this dynamic-programming algorithm applied to the boundaries of Figure 6a is shown in Figure 6b. The CM took approximately 34 seconds to find these boundaries. (These were the boundaries of the closed regions found by the CM implementation of the image segmentation algorithm, as applied to a larger window of the image from which Figure 2 was derived, and for a relatively large value of s^t.) Note that, even though the boundary of each closed region was encoded independently, the parts of the boundaries shared by two regions are almost always encoded in the same fashion. This indicates that the encoding is fairly stable.

To illustrate in more detail, Figure 6c shows the original boundary for one of the larger regions (corresponding to the house in the image). Figure 6d shows the straight-line segments overlaid on the boundaries. The dots represent the first and last boundary points encoded by each line segment. Figure 6e shows the straight-line segments and dots without the boundaries.

Summary

Much work has been done recently on the problem of reconstructing piecewise-smooth surfaces in one or more dimensions, given corrupted samples of the surface [3,5,12,14,17,20,22,26,24,30]. There are several especially difficult aspects to the problem. The first is to determine automatically the appropriate degree of smoothness of the surface as a function of the given data. The second is to determine automatically both the position and order of the discontinuities. The third is to ascertain when such a description is appropriate for the data. We have resolved these difficulties by (1) posing the problem as an optimization problem in which the objective function is based on the information-theoretic notion of minimum-length descriptions, and (2) defining an algorithm that balances simplicity of description against stability of description by first finding the most stable aspects of the description.

We have presented a powerful approach to the image-partitioning problem: *construct a complete and stable description of an image in terms of a descriptive language that is simplest in the sense of being shortest*. We have presented criteria on which to base formal definitions of completeness, stability, and simplicity, and we have embodied these criteria within the theory of minimum-length descriptions.

For the specific image-partitioning problem, we described real images as the corruption of ideal (piecewise-polynomial) images by blurring and the addition of spatially varying white noise. We defined a language for describing both the ideal image and the corruptions, and presented an algorithm for finding the simplest description of an image, in terms of this language, for a given measure of stability. This algorithm is a significant improvement over that presented in [16]. The stability measure has proved *crucial* because we are interested in descriptions that are not only as simple as possible, but that are also as invariant as possible to the severe approximations embodied in any low-level descriptive language. The algorithm not only determines the position of discontinuities in the ideal image, but also determines both the order of the discontinuity and the order of the polynomial within the regions; all of this is done without the need to adjust any parameters. Furthermore, the algorithm is local, parallel, and iterative, making it ideally suited to massively parallel computer architectures such as the CM.

We have also presented preliminary results on the second stage of a proposed hierarchical scheme for describing the image. In this second stage, the region boundaries found by the first stage are described in terms of straight-line segments; the minimal-length description of the boundaries was found using a parallel dynamic-programming algorithm.

Applications of this formalism to real images indicate that, even though the descriptive language we have defined is extremely simple (with no models of three-dimensional shape, lighting, or texture, for example), the simplest and most stable description in this language yields excellent image partitions.

References

[1] A. R. Barron and T. M. Cover. Convergence of logically simple estimates of unknown probability densities. In *1983 International Symposium on Information Theory*, St. Jovite, Québec, Canada, 1983.

[2] H. G. Barrow and J. M. Tenenbaum. Recovering intrinsic scene characteristics from images. In *Computer Vision Systems*, pages 3–26, Academic Press, New York, New York, 1978.

[3] P. J. Besl and R. C. Jain. Segmentation through variable-order surface fitting. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 10:167–192, 1988.

[4] A. Blake. Comparison of the efficiency of deterministic and stochastic algorithms for visual reconstruction. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 11:2–12, 1989.

[5] A. Blake and A. Zisserman. *Visual Reconstruction*. MIT Press, Cambridge, Massachusetts, 1987.

[6] J. F. Canny. A computational approach to edge detection. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 8:679–698, 1986.

[7] P. Cavanagh. Reconstructing the third dimension: interactions between color, texture, motion, binocular disparity, and shape. *Journal of the ACM*, 1988.

[8] G. J. Chaitin. Algorithmic information theory. *IBM Journal of Research and Development*, 21:350–359, 1977.

[9] G. Dahlquist and Å. Björck. *Numerical Methods*. Prentice Hall, Englewood Cliffs, New Jersey, 1974.

[10] S. Geman and D. Geman. Stochastic relaxation, gibbs distributions, and the bayesian restoration of images. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 6:721–741, 1984.

[11] M. P. Georgeff and C. S. Wallace. *A General Selection Criterion for Inductive Inference*. Technical Note 372, SRI International, Menlo Park, California, 1985.

[12] W. E. L. Grimson and T. Pavlidis. Discontinuity detection for visual surface reconstruction. *Computer Vision, Graphics, and Image Processing*, 30:316–330, 1985.

[13] D. H. Hubel and M. S. Livingstone. Segregation of form, color, and stereopsis in primate area 18. *Journal of Neuroscience*, 7:3378–3415, 1987.

[14] D. J. Langridge. Detection of discontinuities in the first derivatives of surfaces. *Computer Vision, Graphics, and Image Processing*, 27:291–308, 1984.

[15] Y. G. Leclerc. Constructing simple stable descriptions for image partitioning. *International Journal of Computer Vision*, 3(1), 1989. in press.

[16] Y. G. Leclerc. Constructing simple stable descriptions for image partitioning. In *Proceedings of the 1988 DARPA Image Understanding Workshop*, pages 365–382, DARPA, Cambridge, Massachusetts, April 1988.

[17] D. Lee and T. Pavlidis. One-dimensional regularization with discontinuities. In *Proceedings of the First International Conference on Computer Vision*, pages 572–577, London, England, June 1987.

[18] D. G. Luenberger. *Linear and Nonlinear Programming.* Addison-Wesley, Menlo Park, California, second edition, 1984.

[19] D. Marr. *Vision.* W. H. Freeman, San Francisco, California, 1982.

[20] J. Marroquin, S. Mitter, and T. Poggio. Probabilistic solution of ill-posed problems in computational vision. *Journal of the American Statistical Association*, 82:76–89, 1987.

[21] M. L. Minsky. Problems of formulation for artificial intelligence. In R. E. Bellman, editor, *Mathematical Problems in the Biological Sciences, Proceedings of Symposia in Applied Mathematics XIV*, page 35ff, American Mathematical Society, Providence, Rhode Island, 1962.

[22] D. Mumford and J. Shah. Boundary detection by minimizing functionals, I. In *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pages 22–26, San Francisco, California, June 1985.

[23] E. P. D. Pednault. Inferring probabilistic theories from data. In *Proceedings of the Seventh National Conference on Artificial Intelligence*, pages 624–628, Saint-Paul, Minnesota, 1988.

[24] E. P. D. Pednault. Some experiments in applying inductive inference principles to surface reconstruction. In *Proceedings of the Eleventh International Joint Conference on Artificial Intelligence*, 1989. (submitted).

[25] J. Rissanen. A universal prior for integers and estimation by minimum description length. *The Annals of Statistics*, 11:416–431, 1983.

[26] P. Saint-Marc and G. Medioni. Adaptive smoothing for feature extraction. In *Proceedings of the 1988 DARPA Image Understanding Workshop*, pages 1100–1113, Boston, Massachusetts, April 1988.

[27] G. B. Smith and H. C. Wolf. *Image-to-Image Correspondence: Linear-Structure Matching.* Technical Note 331, SRI International, July 13 1984.

[28] R. J. Solomonoff. A formal theory of inductive inference. Parts I and II. *Information and Control*, 7:1–22,224–254, 1964.

[29] R. Sorkin. A quantitative Occam's razor. *International Journal of Theoretical Physics*, 22:1091–1103, 1983.

[30] D. Terzopoulos. Regularization of inverse visual problems involving discontinuities. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 8:413–424, 1986.

[31] A. W. Witkin. Scale-space filtering. In *Proceedings of the Eighth International Joint Conference on Artificial Intelligence*, pages 1019–1021, Karlsruhe, West Germany, 1983.

[32] A. W. Witkin, D. Terzopoulos, and M. Kass. Signal matching through scale space. *International Journal of Computer Vision*, 1:133–144, 1987.

[33] S. W. Zucker. Early orientation selection: tangent fields and the dimensionality of their support. In T. Caelli and P. Dodwell, editors, *Figural Synthesis*, Lawrence Erlbaum Associates, Hillsdale, New Jersey, 1984.

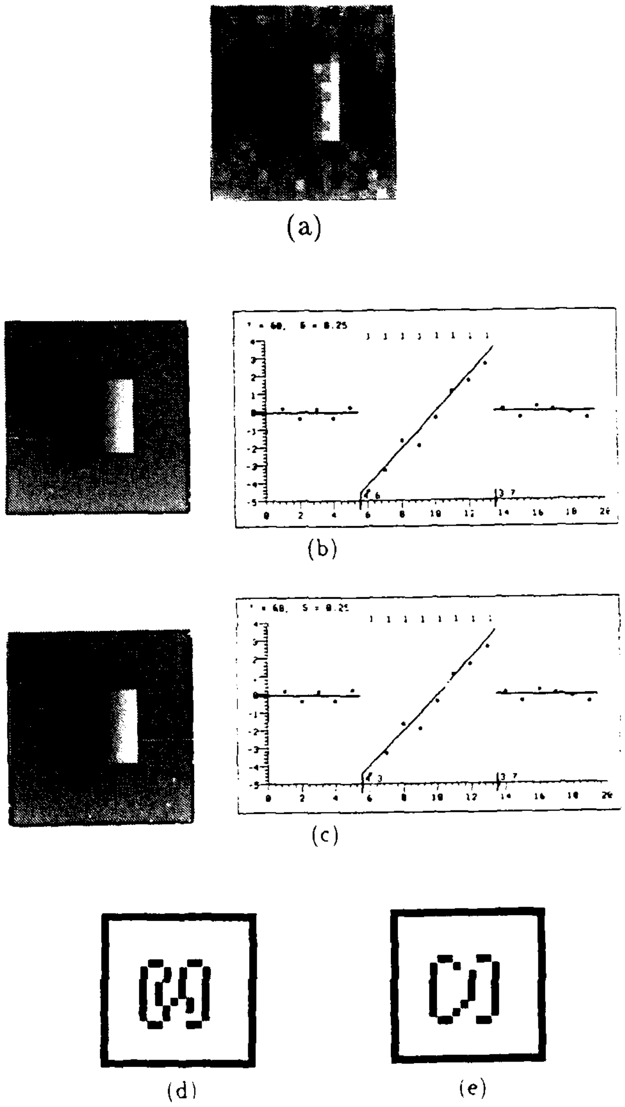


Figure 1: An illustration of the power of global optimization. (a) The input synthetic image. (b) The result of the procedure for $p_{max} = 1$. (c) The result of the procedure for $p_{max} = 2$. (d) The output of the Canny operator, mask size=4. (e) The output of the Canny operator, mask size=8.

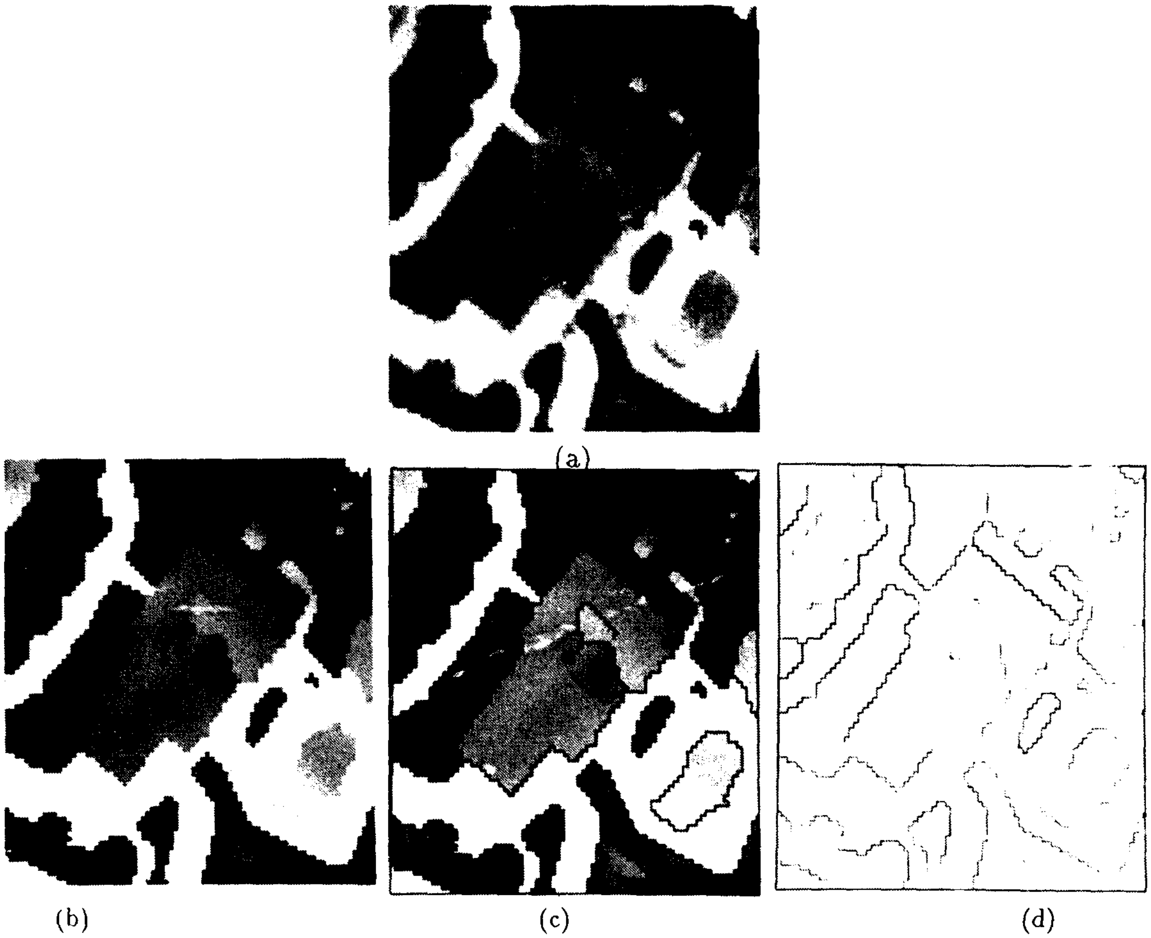


Figure 2: An application of the procedure to an aerial image of a house, with $p_{max} = 1$. (a) The input image. (b) The resulting underlying image. (c) The underlying image with overlaid discontinuities. (d) The stability measure of the discontinuities; the darkest discontinuities are the most stable.

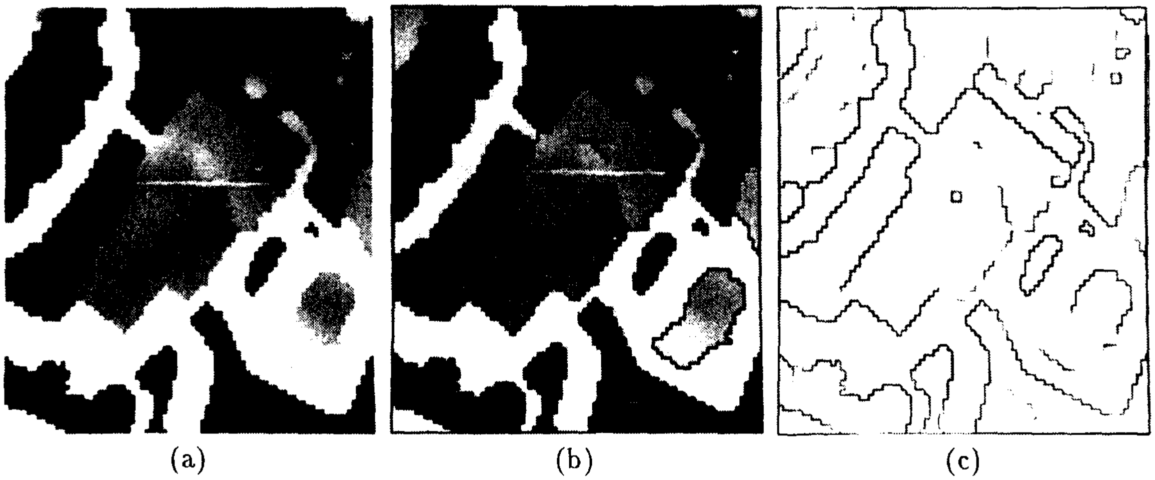
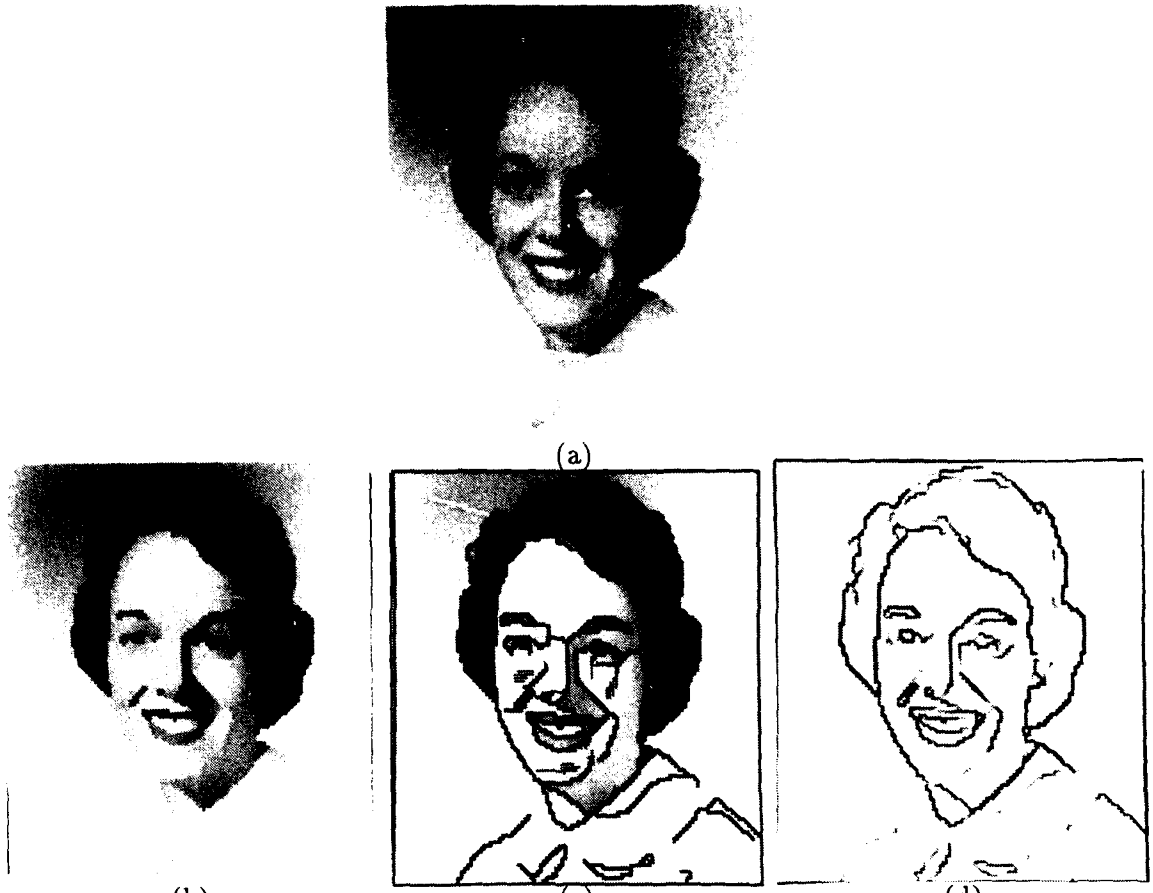


Figure 3: Same as the prior figure, but with $p_{max} = 2$. (a) The resulting underlying image. (b) The underlying image with overlaid discontinuities. (c) The stability measure of the discontinuities.


(a)

(b) (c) (d)

Figure 4: An application of the procedure to the image of a face, with $p_{max} = 1$. (a) The input image. (b) The resulting underlying image. (c) The underlying image with overlaid discontinuities. (d) The stability measure of the discontinuities.

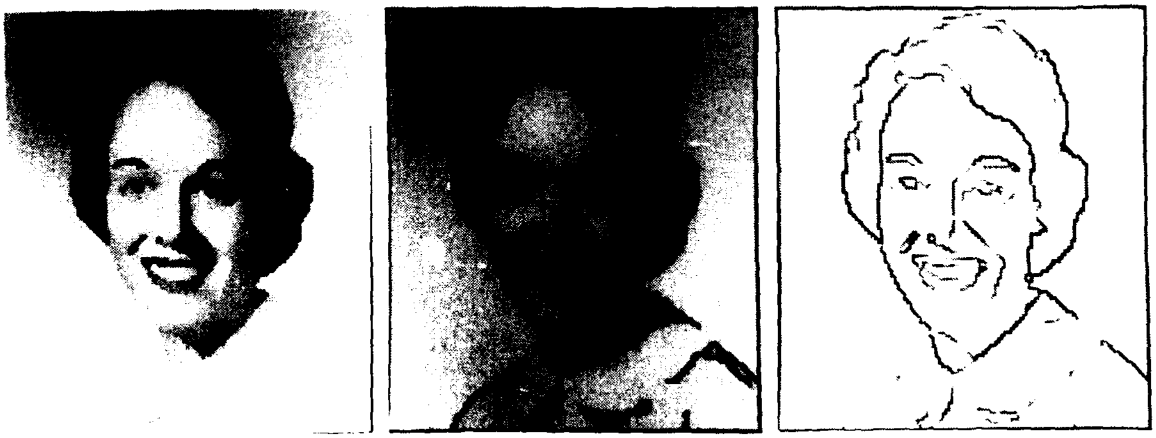

(a) (b) (c)

Figure 5: Same as the prior figure, but with $p_{max} = 2$. (a) The resulting underlying image and discontinuities. (b) The underlying image with overlaid discontinuities. (c) The stability measure of the discontinuities.

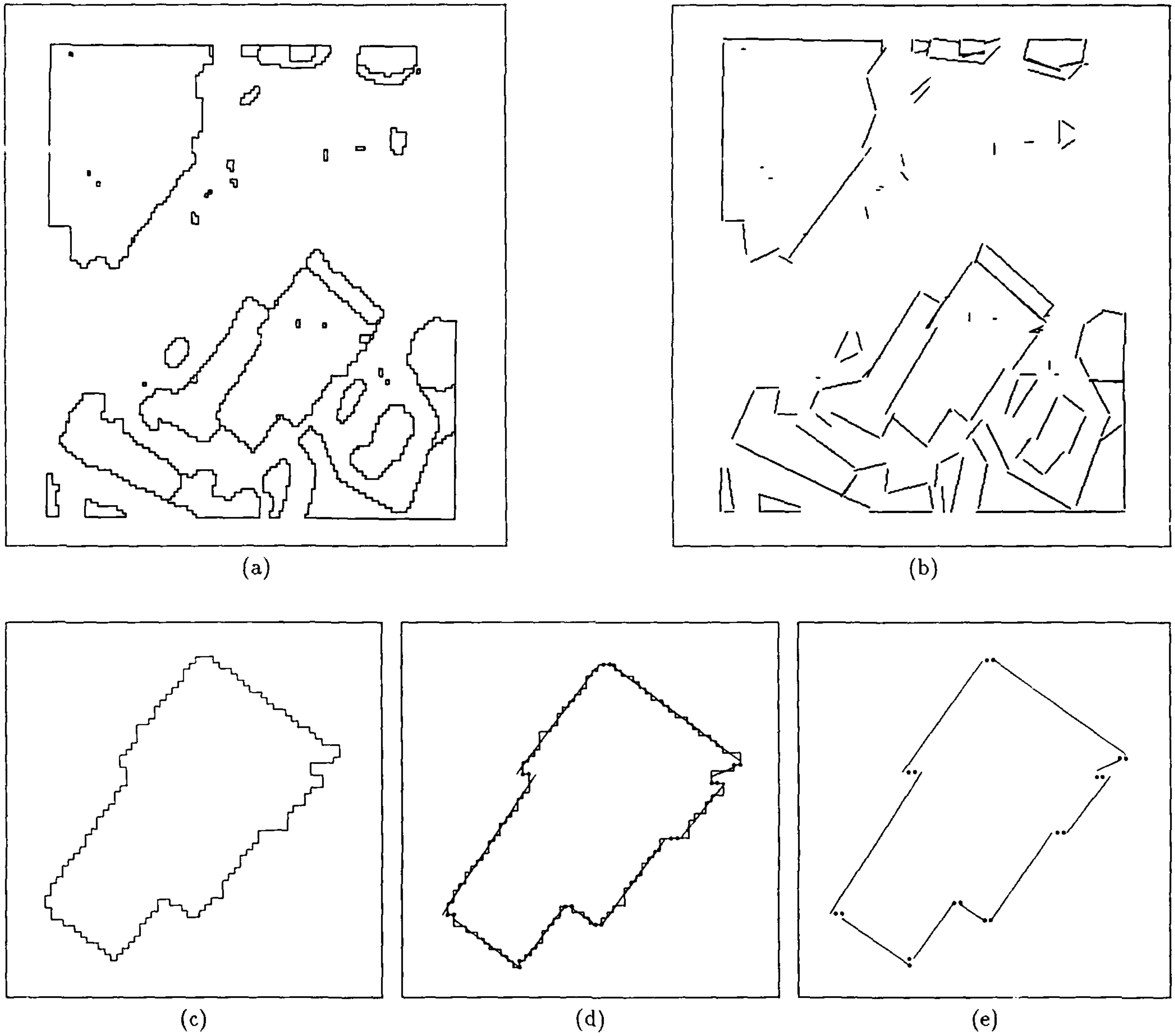


Figure 6: Result of the minimal-encoding of region boundaries in terms of straight lines using dynamic programming. (a) The initial boundaries. (b) The straight-line segmentation. (c) The boundary of the house only. (d) The straight-line segmentation of the house superimposed on the boundary. The dots represent the first and last points encoded by each line segment. (e) The straight-line segmentation of the house only.

Edge Based Transform Refinement [1]

D.W. Thompson

GE Corporate Research and Development Center
Schenectady, NY 12301

ABSTRACT

The problem of model-based feature matching is complicated by the need for robustness in the face of segmentation error and the need for a form of hypothesis verification. A fast solution to the problem of determining the transformation of a set of perturbed edge segments is presented as a partial solution to both these problems. This solution is also seen as an approach to the object tracking problem when transformations between successive images are relatively small.

INTRODUCTION

It is often the case that in order to increase the robustness of a model-based matching system one is forced to allow considerable flexibility in match parameters. This allows for considerable image variation with a reasonable chance of successfully locating the objects of interest. However, one also incurs the penalty of reducing the specificity of the match. This loss of specificity is manifested in a reduced numerical precision of the match and by an increase in the number of false positive results. The rejection of false positive matches and acceptance of correct matches is hypothesis verification. Hypothesis verification may use the same or different data modalities than the original hypothesis generation algorithm (eg. edges rather than vertices).

Our need for a match refinement procedure is motivated by the characteristics of our vertex-pair matching system [Thompson and Mundy]. First, due to the nature of the vertex-pair feature, we are often not able to involve some areas of the model in the matching procedure. In order to behave well, the vertex-pair must be defined on a pair of intersecting edges which are long enough to be readily detected in a wide range of poses. It is not unusual that strong, "interesting" edges exist alone. Second, the vertex-pair matcher derives its robustness from the fact that transform votes are coarsely defined. This allows vertex-pairs to exhibit "matching" transforms despite a certain amount of anomalous image/segmentation behavior, but it also means that the resulting transform values may be imprecise. Third, in the area of object tracking, we need a simple approach to viewpoint-constrained matching. A refined vertex-pair approach has been tried and found to exhibit reasonable matching behavior but to run relatively slowly [Thompson and Mundy 88]. We therefore need an approach to allow coarse matching to occur (to increase the probability of a successful match and to speed the matching procedure) but which may refine the quality and precision of matches in a precise numerical framework.

PRIOR WORK

The primary work on transform determination from 3d to 2d line correspondences was done by Lowe [Lowe]. His system used Newton-Raphson convergence over a set of equations produced by measuring the distance between lines in the scene and projected model lines (the lines produced by back projecting the transformed model onto the image plane). Each line correspondence produced two equations, so the system was able to solve for the transform with as few as three lines.

A significant amount of work has been done on solving the general structure-from-motion problem using line correspondences. This is not the problem being addressed here, as we are using a 3d model and attempting to determine motion, but the approach to the solution via 2d geometrical constraints on 3d transform parameters is similar. Liu and Huang [Liu and Huang] present a linear algorithm for general

[1] Work at GE was supported in part by the DARPA Strategic Computing Vision Program in conjunction with the Army Engineer Topographic Laboratories under Contract No. DACA76-86-C-0007 and the AirForce Office of Scientific Research under contract No. F49620-89-C-0033.

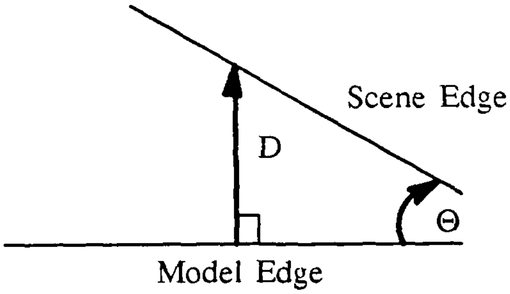


Figure 1: Edge Similarity Measures

motion computation on a series of three synthetic images. Their algorithm dealt with matches between sets of 2d lines in the successive images. They relied on solving for the individual components of three transform matrices, for a total of 27 unknowns, requiring at least 13 line correspondences over three images. Faugeras et al [Faugeras et al] used extended Kalman filtering on a linearized form of edge descriptions to generate a 3d description from a series of images. They require 6 edge correspondences over three images to determine position and orientation.

OUR APPROACH

The approach here is to use the approximate transform derived from an arbitrary matching system (usually the vertex-pair matcher) to predict feature (edge) locations. We then assign scene edges to model edges and compute a differential transform. The new transform is used to predict new edge correspondences, and the process iterates until the system converges.

PREDICTING EDGE CORRESPONDENCES

We must parameterize model and scene edges in such a way that we may match model edges to scene edges which may be broken or over/under extended, yet which have approximately the right orientation and position normal to the model edge projection. The orientation component, Θ, is simply the angular difference between the edges. The position component, D, is more complicated since scene edges may be fragmented by occlusion or segmentation error. Since the model edge projects perfectly, the location of its midpoint is reliable. Therefore, we can use the component of the distance between the scene line (which is the geometric line along the scene edge) and the midpoint of the projected model edge which is normal to the projected model edge. The parameters D and Θ are shown in Figure 1. To determine candidate line correspondences, the model is projected onto the scene and each model edge is paired with every scene edge which falls within permissible ranges of D and Θ. The viewpoint may be compared with model face normals to eliminate lines in the model which are not visible and should not be matched.

DETERMINING A DIFFERENTIAL TRANSFORM

In order to determine a differential transform, we consider the set of edge correspondences, as derived in the last section, to define a linear system of equations. We assume affine projection as an adequate approximation of true perspective viewing [Thompson and Mundy]. Since the transformation equations are not linear, we need to linearize the rotation equations of line correspondences.

Determining Rotation

If we consider R to be the complete rotation matrix and let Rb be the initial rotation estimate and Rt be the differential rotation matrix, we can write $R = Rb \cdot Rt$. If we only allow small rotations, Rt becomes a linear transform in ϕ, ψ, ζ, rotation about the x, y and z axes respectively. Specifically,

$$R = \begin{Vmatrix} \cos\zeta\cos\psi & \cos\zeta\sin\psi\sin\phi - \sin\zeta\cos\phi & \cos\zeta\sin\psi\cos\phi + \sin\zeta\sin\phi \\ \sin\zeta\cos\psi & \sin\zeta\sin\psi\sin\phi + \cos\zeta\cos\phi & \sin\zeta\sin\psi\cos\phi - \cos\zeta\sin\phi \\ -\sin\psi & \sin\phi\cos\psi & \cos\phi\cos\psi \end{Vmatrix}$$

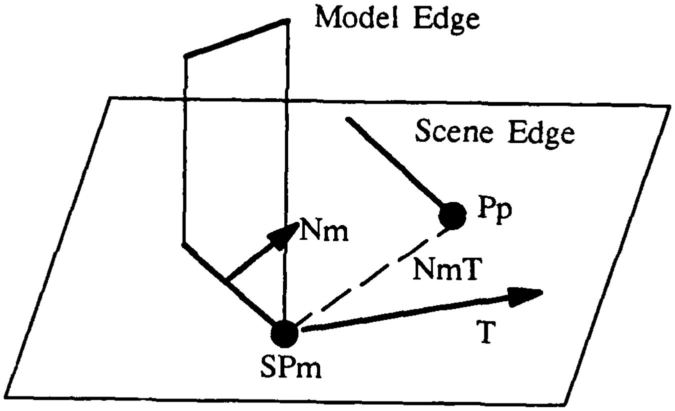


Figure 2: Determining Scale and Translation.

Using the standard small angle substitution for sin and cos (cos $\alpha = 1$, sin $\alpha = \alpha$ for $\alpha \sim 0$),

$$Rt = \begin{Vmatrix} 1 & \psi\phi - \zeta & \psi + \zeta\phi \\ \zeta & \zeta\psi\phi + 1 & \zeta\psi - \phi \\ -\psi & \phi & 1 \end{Vmatrix} \tag{1}$$

We may further simplify Rt by noting that for small rotations, the product of two angles is very small (≈ 0). Therefore let

$$Rt = \begin{Vmatrix} 1 & -\zeta & \psi \\ \zeta & 1 & -\phi \\ -\psi & \phi & 1 \end{Vmatrix} \tag{2}$$

Let Ns be the normal vector to the 2d scene edge and M be the original model edge direction vector. When perfectly rotated, M should be orthogonal to Ns. Thus we can apply a dot product constraint,

$$Ns \cdot R \cdot M = 0 \tag{3}$$

$$Ns \cdot Rt \cdot Rb \cdot M = 0 \tag{4}$$

$$Ns \cdot Rt \cdot Nb = 0 \tag{5}$$

substituting Nb for the approximately rotated 3d edge ($Rb \cdot M$) in eq. 4.

$$Rt \cdot Nb = \langle Nb_1 - \zeta Nb_2 + \psi Nb_3, \zeta Nb_1 + Nb_2 - \phi Nb_3, -\psi Nb_1 + \phi Nb_2 + Nb_3 \rangle \tag{6}$$

Substituting eq. 6 into eq. 5 (recall Ns is 2 dimensional)

$$Ns_1 Nb_1 + Ns_2 Nb_2 - \phi Ns_2 Nb_3 + \psi Ns_1 Nb_3 + \zeta(Ns_2 Nb_1 - Ns_1 Nb_2) = 0 \tag{7}$$

or,

$$Ns_1 Nb_1 + Ns_2 Nb_2 = \phi Ns_2 Nb_3 - \psi Ns_1 Nb_3 - \zeta(Ns_2 Nb_1 - Ns_1 Nb_2) \tag{8}$$

Each line correspondence provides a single equation. The system solves for the three rotation parameters, ϕ, ψ, ζ, by a system of linear equations. We therefore need a minimum of three line correspondences. If more than three line correspondences exist (which is likely) a least squares approximation is used.

Determining Translation and Scale

The next step is to compute translation and an affine scale measure. Given a unit normal vector, Nm, to the projected model line, we may write

$$Nm \cdot T = Nm \cdot (Pp - SPm) \tag{9}$$

where T is the translation vector, S is the scale factor, Pm is the rotated origin point of the model edge and Pp is the origin point of the scene edge. That is, the component of the translation vector which is normal to the projected model edge is equal to the vector difference between any point on the scene line and any point on the projected model line, which, ideally, are now parallel. Then,

$$Nm \cdot SPm + Nm \cdot T = Nm \cdot Pp \tag{10}$$

and we can solve for S and T. Figure 2 shows the geometry involved. Again, each line correspondence provides one equation and we therefore need three non-parallel lines to minimally determine the three underlying parameters of T and S.

ITERATION CONSIDERATIONS

The computation of a new transform estimate involves computing a transform consistent with the current set of line correspondences. The system finds a consistent transform by iteratively computing a differential transform, from the least mean squares systems described above, and adding it to the current transform. At each iteration, outlying line correspondences are removed. When no outliers remain and the transform parameters are stable, a consistent transform is said to have been reached. Then, the new consistent transform is applied to the model, the model is back-projected, a new set of line correspondences is computed, and the process repeats. The system finishes when the set of line correspondences and the resulting transform remain the same or when it passes a limit on the maximum number of iterations.

RESULTS

The accompanying figures demonstrate the use of the iterative edge matcher for transform refinement and model tracking. Figure 3 shows a wire frame model from a single viewpoint. Figure 4 is an image of the modeled object on a turntable. The vertex-pair matcher was used to generate an initial match, which is shown in fig. 5. Note that the match is correct in certain areas, where the model was parameterized with vertex-pairs, but erroneous in other areas. Figure 6 shows the match after it has been updated by the edge matcher. No face visibility (hidden line removal) criteria were used for this example. Figure 7 shows a second image of the block, rotated 10 degrees on the turntable. Figure 8 shows the match to that image by the iterative edge matcher using the match from fig. 6 as a starting point.

Images A and B are 320 x 240 pixels. Their segmentations each consist of approximately 85 edges. Convergence to the values shown in figs. 6 and 8 given the initial estimates each occurred in under 5 seconds on a Symbolics 3600.

Convergence of the matcher is dependent on the available image data, but has been demonstrated with up to 25^o error in the initial match. Currently, the system is judged to have converged when it settles on a fixed set of line correspondences and computes nearly the same differential transform for them twice in a row. These are usually adequate criteria but instances of oscillation have occurred. Divergence from a reasonable initial estimate has also occurred, usually in cases where a large number of small edges are present in the segmentation or where the viewpoint is such that only limited or coplanar areas of the object are visible. Convergence on a false match has not been observed with reasonable matching parameters.

CONCLUSION

A simple method for determining model transforms from edge correspondences has been presented. The system is relatively robust in the face of segmentation error and occlusion, and is reasonably fast. This algorithm offers solutions to the problems of transform refinement, hypothesis verification and tracking. Most of our experimentation has been oriented towards simple transform refinement. Hypothesis verification is a larger topic for which this paper only offers a small tool. Work remains to be done on the subject of what constitutes a valid hypothesis. Tracking seems to be a reasonable application for the system discussed here, but a good deal of work remains to be done in generating models of object motion. Ongoing work on the system is intended to account for perspective distortion.

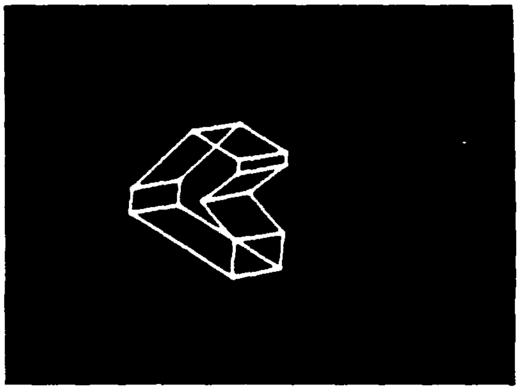


Fig. 3 Wire Frame Model

Fig. 4 Image A

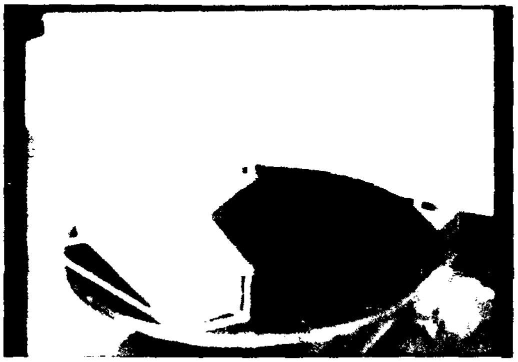


Fig. 5 Initial Match

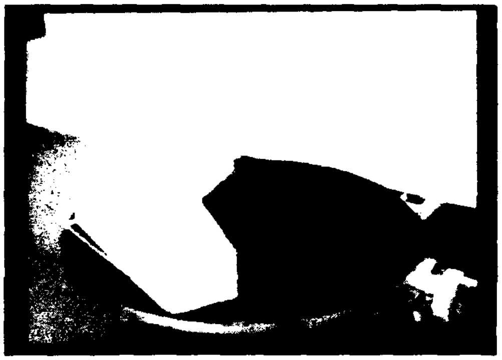


Fig. 6 Updated Match

Fig. 7 Image B

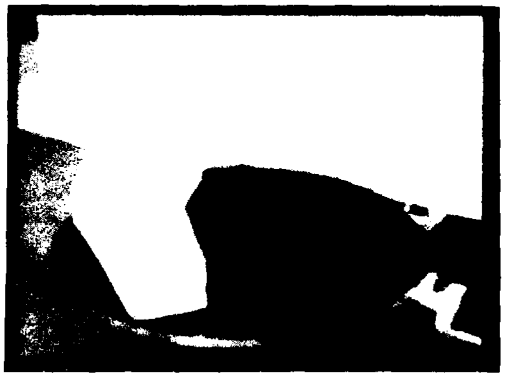


Fig. 8 Edge Match to Image B

References

[Thompson and Mundy 88] Thompson, D.W. and J.L. Mundy,"Model-Based Motion Analysis", Proc. 4th International Symposium on Robotics Research, MIT Press, 1988.

[Liu and Huang] Liu, Y. and Huang, T.S., "A Linear Algorithm for Motion Estimation Using Straight Line Correspondences", Computer Vision, Graphics and Image Processing, 44, 1988, pp. 35.

[Faugeras et al] Faugeras, O., F. Lustman and G. Toscani,"Motion and Structure from Motion from Point and Line Matches", Proc. 1st International Conf. on Computer Vision, 1987, pp. 25.

[Thompson and Mundy] Thompson, D., Mundy, J.L., "Three-Dimensional Model Matching From an Unconstrained Viewpoint", Proc. IEEE Robotics and Automation, 1987, pp. 280.

[Lowe] Lowe, D., "Perceptual Organization and Visual Recognition", Kluwer Academic Publishers, Boston, MA, 1985.

Computer Analysis of Regular Repetitive Textures

Leonard G. C. Hamey[1]
Takeo Kanade
School for Computer Science
Carnegie Mellon University
Pittsburgh PA 15213

Abstract

Regular repetitive textures are common in real-world scenes, occurring in both natural and man-made environments. Their analysis is important for image segmentation and for shape recovery from surface texture. There are two fundamental problems in analyzing regular repetitive texture. Firstly, the frequency interpretation of any regular texture is ambiguous since there are many alternative interpretations that correspond to the same texture. Secondly, the very definition of regular repetition is circular since the texture element and the repetitive frequency are defined in terms of each other. In this paper, we address these two problems and present an answer to each. To address the ambiguity of frequency interpretation we turn to the lattice theory and choose successive minima as the most fundamental frequency vectors of the texture. To deal with the circular definition of regular repetition, we compare the structural relationships to prominent features in the texture. These theoretical concepts are incorporated into a working system, capable of analyzing and segmenting regul^r repetitive textures in real-world images. In contrast with previous work, our technique involves entirely local analysis and is thereby robust to texture distortion.

1 Introduction

Regular repetitive textures are common in real-world scenes. They occur both as a result of natural processes (e.g. the repetitive texture of reptile skin) and the efforts of man (e.g. man's making of a city scene). Understanding these textures is important not only as a basis for image segmentation but also because regular repetitive textures can provide valuable information for estimating surface orientation.

A fundamental problem in analyzing regular textures, however, is that the definition of regular repetitive texture is circular. The frequency of the texture is defined as the spatial displacement between elements of the texture, but the element of the texture is defined as that portion of the image that is regularly repeated. This circular dependency is usually handled by obtaining information about the repetitive frequency without considering the nature of the texture element or vice versa. In both approaches, a global analysis of the texture is performed, restricting the applicability of the algorithms to undistorted samples of a single repetitive texture.

In contrast to these approaches, our work employs a purely local analysis to identify the repetitive structure of the most prominent (dominant) features in regular repetitive textures in real-world images. In this way, we identify the regular repetitive relationships between texture elements without identifying the texture elements themselves.

A second problem in analyzing regular repetitive textures is that: even when we know the locations of texture elements, there are many alternative pairs of frequency vectors that equally describe the pattern of the texture

[1] Current address: Computing Discipline, Macquarie University, NSW 2109, Australia

elements. This problem has been handled in the past by choosing either the shortest frequency vectors [8,14,15] or those vectors which provide the simplest grammatical structure for the texture region [10]. The former approach has the heuristic advantage that the frequency vectors are independent of the shape of the texture region. In this paper, we develop additional reasons for preferring the shortest frequency vectors (which we call the *fundamental frequency vectors* based on results from lattice theory. These results provide additional properties of the fundamental frequency vectors that are useful for extracting the frequency of real-world textures.

We have applied the concepts developed in this paper to a system that analyzes regular repetitive textures in real-world images. This system identifies the fundamental frequency relationships between the most prominent features in the texture elements of regular repetitive textures present in the im ;es. The output of the system is a planar graph describing in detail the grid structure of the texture.

2 Existing Frequency-Based Approaches

Many of the existing approaches for analysis of regular repetitive textures are based on first determining the frequency of the repetition. For this purpose, Matsuyama *et al.* [9] employ the Fourier power spectrum. They locate peaks in the power spectrum as evidence of the dominant frequency of the texture. Other researchers [2,3,17] observe that the co-occurrence matrix for displacement δ should be highly diagonal if δ is a frequency of a regular repetitive texture. In related work, Davis *et al.* [4] suggest computing co-occurrence relationships between image features instead of between raw pixel intensities. Nevatia *et al.* [11] extract repetitive frequency by computing co-occurrences of edge features.

The frequency-based approaches all rely upon a global analysis of the image. They have been successfully employed for extracting repetitive frequency from small samples of a single repetitive texture. If these approaches were to be applied to real-world scenes, however, two problems would have to dealt with. Firstly, real-world scenes often contain regular repetitive textures in which the frequency varies throughout the texture. Secondly, real-world scenes usually contain regions of non-regular texture and often have more than one regular texture in them.

Both of these problems imply that an analysis needs to consider small samples of the image (of the order of a few texture elements) at a time. Because the size of the texture elements is a function of the frequency of the repetition, the appropriate image sample size varies as a function of the hypothesized repetitive frequency. It therefore becomes necessary to process a large number of samples of different sizes instead of simply applying a global analysis to a single image. It is not surprising then that these techniques have yet to be applied to real-world images.

3 Existing Element-Based Approaches

Regular repetitive texture can also be analyzed by first locating the texture elements and then determining the frequency relationships between them. This approach is employed by Tomita *et al.* [14] who extract the texture elements by a simple region analysis. The elements are then grouped on the basis of their shape and orientation parameters. The displacements between the texture elements of a single group are then entered into a two-dimensional histogram. Peaks in the histogram are identified as frequency vectors of the texture. Further work in the same vein is in [8,10].

These element-based approaches have been applied with considerable success to small samples of a single regular repetitive texture. Because the texture elements are grouped, regions of different regular textures can be segmented from each other provided that their texture elements are sufficiently different. This capability is very useful when dealing with real-world images where it is common to encounter more than one regular repetitive texture in the same image.

The element-based approaches do have one major weakness: the repetitive frequency is determined by a global analysis of the texture. Such a global analysis will fail if the frequency varies within the texture. Frequency variations commonly occur in textures in real-world images as a result of perspective imaging, and existing techniques are unable to analyze such images.

4 The Dominant Feature Assumption

Rather than working with the raw image intensities on the one hand or texture elements on the other, we work with features of the image in some arbitrary feature space. We consider the image to consist of a set of these features, appropriately located. (Conceptually, we convert the image into its feature-space representation). We define a texture element T as a set of features $\{F_1, F_2,...,F_n\}$ each of which has a location $\mathbf{x}(F_i)$ relative to the origin of T. We assume that a feature does not span two or more texture elements, but allow a texture element to span any number of features.

We now define a function P, called the *prominence* function, where $P(F_i)$ is a scalar value. The function P is continuous, in that similar features have similar values of P. A convenient form for P is a function that measures information content of the feature. It is highly unlikely that any two features in T will have the same prominence value P. In fact, for a randomly generated T, all values of P will be distinct with probability 1. It follows that we can identify any particular feature by its prominence value P. In particular, we can uniquely identify one of the features of T, say F_p, as the most prominent feature in T. Consider the following:

$$\forall i \; P(F_p) \geq P(F_i)$$

We call F_p the *dominant feature* of the texture element T. Notice that if P is an information measure, then F_p is that feature in T which contains the most information. For this reason, the feature with the largest value of P is selected as the dominant feature.

Consider now a regular repetitive texture consisting of texture elements $T_1, T_2,...,T_m$ that are identical copies of T. If we analyze this texture and extract the features $F_{11}, F_{12},...,F_{nm}$, we will find that there is a group of features F_{P1}, $F_{P2},...,F_{Pm}$ that are all equally prominent and are more prominent than any other features in the texture. The *dominant feature assumption* states that such a set of dominant features always exists.

- **Dominant Feature Assumption:** Every regular repetitive pattern exhibits some feature that occurs once in each texture element and is more prominent than any other feature occurring in the same texture element.

Under the dominant feature assumption, it is possible to determine the structure of regular repetitive textures from the feature-space representation of the texture. Since there is one dominant feature for each texel, the structure of the dominant features captures the structure of the texels, as illustrated in figure 1.

5 Fundamental Frequency Vectors

The fundamental frequency vectors are defined as the shortest pair of frequency vectors in the texture. They have a number of useful properties that can be derived from lattice theory. In this theory, we assume that we are dealing with an undistorted regular repetitive texture. Undistorted regular repetitive textures are closely related to lattices. We define an undistorted regular repetition as follows.

- **Definition 1:** A regular repetition R is a set of points $\{\mathbf{x}_0+i\mathbf{u}+j\mathbf{v}: i, j \text{ integers}\}$ in the plane where $\mathbf{x}_0$ is an arbitrary point and **u** and v are linearly independent vectors. R is the translate of a lattice. The vectors **u** and v are a basis for R.

We define the fundamental frequency vectors **u**′ and v′ of a regular repetition R as the shortest and second-shortest linearly independent frequency vectors of R.

- **Definition 2:** More formally, let $V=\{\mathbf{x}_1-\mathbf{x}_2 : \mathbf{x}_1,\mathbf{x}_2 \in \mathbf{R}; \mathbf{x}_1 \neq \mathbf{x}_2\}$ be the set of all possible frequency vectors of R. Define the first fundamental frequency vector $\mathbf{u}'$ to be any one of the shortest vectors (measured with Euclidean length) in V. Let $U = \{i\mathbf{u}'; i \text{ integer}\}$. Define the second fundamental frequency vector v′ to be any one of the shortest vectors in $V-U$. The vectors **u**′ and v′ are known as successive minima in lattice theory; they are the fundamental frequency vectors of R.

This definition of the fundamental frequency vectors has some ambiguity in the choice of **u**′ and v′. This ambiguity has three possible sources. Firstly, there is the ambiguity between **u**′ and -**u**′ and between v′ and -v′. Secondly, if $|\mathbf{u}'|=|\mathbf{v}'|$, ambiguity can occur in the labeling of the fundamental frequency vectors. However, both cases are not serious problems and will not concern us further.

The third form of ambiguity occurs when there are two linearly independent candidates for v′. In this case there is more than one set of equally valid fundamental frequency vectors that provide different interpretations of the same regular repetitive pattern. Textures which have this third form of ambiguity are referred to as *ambiguous* regular repetitive textures. The texture in figure 2 is ambiguous -- the fundamental frequency vectors are not uniquely defined and the texture can be equally interpreted as skewed to the right or skewed to the left. This form of ambiguity is an important property of the texture itself.

Working from this mathematical basis, we can prove the following useful properties of the fundamental frequency vectors [6,7,5].

1. The fundamental frequency vectors **u**′ and v′ form a basis for R; i.e. the set $\{\mathbf{x}_0+k\mathbf{u}'+l\mathbf{v}: k, l \text{ integers}\}$ is identical to R.
2. There does not exist for R a pair of basis vectors **a** and **b** for which $|\mathbf{a}|<|\mathbf{v}'|$ and $|\mathbf{b}|<|\mathbf{v}'|$; i.e. there is no basis for R consisting of vectors shorter than the longer fundamental frequency vector.
3. There does not exist a pair of basis vectors **a** and **b** for R for which $|\mathbf{a}|+|\mathbf{b}|<|\mathbf{u}'|+|\mathbf{v}'|$; i.e. **u**′ and v′ describe the minimum-perimeter structural unit parallelogram of R.
4. The fundamental frequency vectors **u**′ and v′ are the most perpendicular basis for R; i.e. $|\mathbf{u}'\cdot\mathbf{v}'| / |\mathbf{u}'||\mathbf{v}'|$ is minimal among all bases of R.
5. $|\mathbf{u}'\times\mathbf{v}'|\leq\frac{1}{2}|\mathbf{u}'|^2$ with equality holding exactly when R is ambiguous.
6. The fundamental frequency vectors **u**′ and v′ are separated by an angle of between 60 and 120 degrees (or between -60 and -120 degrees).

The first of the above properties is simply assurance of the validity of the fundamental frequency vectors as a description of the texture frequency. The remaining properties serve firstly to emphasize the unique qualities of the fundamental frequency vectors --they are uniquely short and perpendicular among all possible frequency descriptions of the texture. These properties are also useful as a basis for identifying fundamental frequency vectors from amongst the many candidate frequency vectors in a texture.

In addition to the above properties of the fundamental frequency vectors, the fundamental frequency vectors are related to the Relative Neighbourhood Graph (RNG) of the texture elements. The RNG [16] is a bidirectional graph on a set of points $P = \{\mathbf{p}_1,\mathbf{p}_2,...,\mathbf{p}_n\}$ in the plane. The RNG connects two points $\mathbf{p}_i$ and $\mathbf{p}_j$ if and only if there exists no other element $\mathbf{p}_k \in P$ such that $|\mathbf{p}_k-\mathbf{p}_i|<|\mathbf{p}_j-\mathbf{p}_i|$ and $|\mathbf{p}_k-\mathbf{p}_j|<|\mathbf{p}_j-\mathbf{p}_i|$. The RNG has been proposed as a model of human perceptual grouping of dot patterns [12]. We have shown in [5] that the RNG of an undistorted regular repetitive texture captures exactly the fundamental frequency relationships between the texture elements. More specifically, if W is the set of all possible fundamental frequency vectors of R (including all forms of ambiguity) then the RNG of R is exactly that graph that connects each point $\mathbf{x}_i \in R$ to all points in the corresponding set $\{\mathbf{x}_i+\mathbf{w}:\mathbf{w} \in W)\}$.

6 Analyzing Real-World Textures

In analyzing real-world textures, our aim is to discover the fundamental frequency vectors between the dominant features of the texture elements. The application of the theoretical concepts of dominant features and fundamental frequency vectors is not direct, however, because of the variations that can occur in real-world textures. Firstly, the texture elements of real-world textures often vary from each other, giving rise to variations in the prominence of the dominant features. Secondly, real-world regular repetitive textures are often distorted as a result of effects such as surface curvature and perspective imaging, causing the results obtained from lattice theory to be violated.

In order to deal with the variations in real-world regular textures, we use the relative dominance of the features rather than searching for absolutely dominant features. Rather than computing the Relative Neighbourhood Graph, we develop a structural description in which various properties of the fundamental frequency vectors are combined to produce a graph with weighted edges. Successive refinement steps lead to a robust algorithm that can reliably extract the fundamental frequency description.

The system that we have implemented uses an algorithm consisting of four major phases. The first phase is feature detection. It extracts blob-like features from the input image and associates a prominence value and image location with each feature. The second phase establishes basic structural relationships between the features, based on the dominant feature assumption and properties of the fundamental frequency vectors. The structural relationships are represented as weighted links between the features. The links and their associated weights are passed to the third phase which constructs locally regular repetitive structures and attaches evaluations to each of them. Multiple candidate repetitive structures are evaluated for each image feature. The final phase decides up on a locally consistent repetitive structure interpretation based upon the competing repetitive structures. A relaxation algorithm is used to obtain consistency by constraint propagation.

Throughout the processing steps of this system, multiple competing hypotheses are maintained and passed to subsequent levels of processing. This allows decision-making to be delayed until more information can be incorporated. Back-tracking is made unnecessary at the expense of maintaining all the likely alternative hypotheses. In practice, we found it necessary to maintain at most 50 alternatives at each stage.

7 Smooth Thresholding

Throughout the algorithms described below, we express evaluations on the range 0.0 to 1.0. These evaluations have a similar role to fuzzy reasoning or probability values. Normal decision-making often takes the form of thresholding applied to a measured value. Instead of directly thresholding the data, we employ "smooth thresholding" functions that convert measured values into evaluations on the range 0.0 to 1.0. The smooth thresholding functions T and T_L

are based on the *tanh* function. In the following equations, τ is the nominal threshold value such that $T(\tau,\tau,\sigma)$ is 0.5, and σ is the spread such that $T(\tau+\sigma,\tau,\sigma)$ is 0.75. T is a particular parameterization of the Sigmoid distribution; T_L is a logarithmic version which is more appropriate for evaluating ratios.

$$T(x,\tau,\sigma) = \frac{1}{2} tanh \frac{x-\tau}{\sigma/0.55} + \frac{1}{2}$$

$$T_L(x,\tau,\sigma) = T(log\frac{x}{\tau}, 0, log\sigma)$$

8 Feature Extraction

The first step in our algorithm is to extract features from the image. Conceptually, the processing can be based on any features that have point locations in the image (e.g. comers, but not edges) and an associated prominence value. In the experiments described in this paper, the feature detector was based on one proposed by Ahuja and Haken [1] which employs Laplacian of Gaussian filtering to identify blob features. The detector measures both the diameter and intensity contrast of the blob features. The prominence of a blob is defined, for our purposes, as its area multiplied by the magnitude of its intensity contrast.

It is important that all the potentially important features are initially extracted. The feature extraction phase produces a large number of features including many that may not be relevant to the final analysis. These features generally have low prominence values and they are effectively disregarded during the later stages of the analysis. The results of feature extraction on figure 5 are shown in figure 6.

9 Basic Structural Relationships

The second phase in our algorithm links together pairs of features that are candidates for being the dominant features in neighboring texture elements (i.e. elements that are related to each other by fundamental frequency vectors). The input data for this phase is the features extracted in the first phase together with their prominence values. The output consists of weighted directional links between each feature and selected neighboring features.

This stage of the processing is based on the dominant feature assumption combined with the property of the fundamental frequency vectors that they are equivalent to the relative neighborhood graph. The theory is employed in two principles that are evaluated for each link under consideration.

1. Prejudice principle: Features prefer to be linked to other features which are equally or more prominent than themselves.

The reasoning behind the prejudice principle is two-fold. First, if the feature under consideration is the dominant feature of a texture element, then it is only interested in being linked to the dominant features of other texture elements. Those dominant features should be equally or more prominent than the feature under consideration. On the other hand, if the feature being considered is not a dominant feature, then it will find in its neighborhood more prominent features. By linking to those more prominent features, information will be available to later determine that it is not a dominant feature. It is not possible at this stage in the processing to determine whether the feature under consideration is or is not a dominant feature of a regular texture.

The logarithmic smooth threshold function is incorporated into the following equation S_R which evaluates the directional link between F_i and F_j. The evaluation is suppressive, in that large values of S_R weaken the link. The constants 8 and 1.5 are the values used in our experiments.

$$S_R(F_i,F_j)=T_L(\frac{P(F_i)}{min(P(F_i),P(F_j))},8,1.5)$$

2. Interference Principle: The link between two features is only strong if there is no other equally or more prominent feature within the *dominated region* of the two features.

The interference principle captures concepts derived from both the dominant feature assumption and the relationship between the fundamental frequency vectors and the RNG. Recall that the RNG links together pairs of points where there is no other point point closer to both points under consideration. This is equivalent to linking pairs of points when there is no other point within the "lune" of the two points under consideration (the "lune" is the intersection of two circles as shown by the dotted line in figure 3).

In adapting the RNG result to real-world image data, we must allow for slight discrepancies in the positions of the dominant features, particularly as a result of distortion of the texture. The "lune" is much too sensitive to small variations in the positions of the features and the "true" fundamental frequency vectors would often be disallowed, so we use a region that lies within the "lune". We call this region the dominated region of the two features. As with the application of the prejudice principle we do not simply make a binary decision about whether a particular feature lies inside or outside the dominated region of other features. Rather, the interference of a feature increases from 0 towards 0.5 as the feature approaches the center of the dominated region. Figure 3 shows some contours of the dominated region function.

In our algorithm, we are interested in linking not points, but dominant features. Clearly, it is acceptable for less prominent (and therefore, not dominant) features to occur between the two features under consideration, but any equally prominent or more prominent features would indicate that the pair of features are not neighboring dominant features. The dominated region suppression term (which is larger if the dominated region contains highly prominent features that "interfere" with the link) is given by the following equations.

$$S_I(F_i,F_j)=1-\prod_k [\ 1-S_L(F_i,F_j,F_k)\,S_P(F_i,F_j,F_k)\ ]$$

where

$$S_P(F_i,F_j,F_k)=T_L(\frac{P(F_k)}{min(P(F_i),P(F_j))},0.5,1.5)$$

$$S_L(F_i,F_j,F_k)=\frac{1}{2}-\frac{1}{2}T(\frac{y^2}{(x-1/2)^2\,(x+1/2)^2},1.25,1.25)$$

$$x=(\frac{D(F_i,F_k)}{D(F_i,F_j)})^2-(\frac{D(F_i,F_k)}{D(F_i,F_j)})^2)/2$$

$$y^2=(\frac{D(F_i,F_k)}{D(F_i,F_j)})^2-(x+\frac{1}{2})^2$$

In these equations, $D(F_i, F_j)$ represents the Euclidean distance between the features F_i and F_j.

The link evaluation E_L is obtained multiplicatively from the suppression terms as follows.

$$E_L(F_i,F_j) = (1 - S_R(F_i,F_j))(1 - S_I(F_i,F_j))$$

10 Local Repetitive Structures

The output of the second phase of the algorithm is weighted links between features. The link weights reflect the constraints that the linked feature be highly prominent and that there be no more prominent features in the dominated region of the link. These constraints are not sufficient to ensure that the links are fundamental frequency vectors of regular repetitive textures. Indeed, links will be generated even in random patterns of features. It is the task of this third phase of processing to identify local repetitive structures consisting of a number of links that are indicative of a small piece of regular repetitive texture.

Three local repetitive structures are considered. Within the interior of a texture region, cross (+)-shaped structures of five texture elements can be expected to be found. Along the boundaries, T-shaped structures of four texture elements will occur and at corners, L-shaped structures of three texture elements. These three structures are illustrated in figure 4. The links from phase two are combined into local groupings of +, T and L-shaped arrangements. Each grouping is then evaluated to determine whether it is a candidate for a piece of regular repetitive texture.

In evaluating local repetitive structures, we consider the requirement that the fundamental frequency vectors are to be approximately perpendicular to each other. Certainly, they are to lie within angle of 60 to 120 degrees, or -60 to -120 degrees of each other. The perpendicularity of two vectors **u** and **v** is evaluated by applying a smooth threshold to the squared cosine of the angle between the vectors. In T-shaped and +-shaped structures, 2 and 4 squared cosines are computed respectively and averaged. The average squared cosine S is converted to an evaluation of perpendicularity Q that is multiplied together with other terms to give the overall evaluation.

$$Q = 1 - T_L(S, 0.5, 2.25)$$

In +-shaped and T-shaped repetitive structures, we evaluate the deviation of the structure from perfect regularity. We consider the subsets of three features contained in the structures that should be colinear and equally spaced if the structure is, in fact, regular repetition. There are two such subsets in +-shaped structures and one subset in the T-shaped structures. The deviation from regularity of such a subset is a two-dimensional vector $\mathbf{x}_e$ that is the discrepancy between the average location of the two outside features and the location of the central feature. This discrepancy is interpreted under a pseudo variance-covariance matrix derived from the vectors of the repetitive structure. This yields a modified Mahanalobis distance E^2 which is smooth-thresholded to yield the deviation evaluation D as follows.

$$D = T_L(\frac{1}{E^2}, 4, 1.5)$$

We also consider the evaluations of the individual links contributing to the repetitive structure. For T-shaped and L-shaped structures, bias terms of 0.5 and 0.25 respectively are multiplicatively introduced to compensate for the lower number of contributing links and the reduction of other constraints such as deviation from perfect repetition. The final evaluation of a hypothesized repetitive str:icture is the product of the perpendicularity term Q, the deviation term(s) D, the bias term and the individual link evaluations.

11 Relaxation

After phase three of the algorithm, we have computed many competing regular repetitive structure hypotheses for each feature in the image. If a particular feature is a dominant feature of a regular repetitive texture, we expect that neighboring dominant features will also exist and that these neighboring features will have compatible hypotheses about the local repetitive structure. This constraint that neighboring interpretations should be consistent is implemented in a relaxation algorithm that determines at most one repetitive structure for each texture element. In the e vent that a feature is not participating in a regular repetitive structure, there is a lack of support from neighboring features and the relaxation algorithm determines that no repetitive structure is present.

Phase four commences by discarding any local repetitive structures that have evaluations less than 0.5 for +-shaped structures, 0.25 for T-shaped and 0.125 for L-shaped. (The different threshold levels are required by the bias values introduced in phase three). After the weak hypotheses have been removed, relaxation is applied to strengthen the hypotheses that receive support from their neighboring features. The relaxation algorithm we employ is as follows.

For each feature F_i, each of the associated hypothesized repetitions R_{ij} is examined in turn. Each hypothesis specifies two, three or four neighboring features F_{ijk}. Let $R^*{}_{ijk}$ denote the highest valued hypothesis at feature F_{ijk}. The relaxation strengthens R_{ij} if it is compatible with $R^*{}_{ijk}$. Let $V(R_{ij})$ denote the current evaluation of R_{ij}, $V'(R_{ij})$ denote the new evaluation and $M(R_{ij}, R^*{}_{ijk})$ be the compatibility evaluation for R_{ij} and $R^*{}_{ijk}$.

$$V(R_{ij}) = B(V(R_{ij}), M(R_{ij}, R^*{}_{ijk})), \tau_B, V(R^*{}_{ijk}))$$

$$B(e_1, m, \tau_B, e_2) = \frac{e_1[\tau_B(1-e_2)+me_2]}{e_1[\tau_B(1-e_2)+me_2] + \{\tau_B(1-e_1)}$$

This update rule is applied to each of the neighboring features F_{ijk} in turn. The value of τ_B was 0.5 in our experiments. The compatibility of a pair of hypotheses is evaluated by matching their constituent frequency vectors in a manner similar tc the deviation evaluation in phase three above.

12 Results

The system described in this paper has been applied to 40 images from a variety of real-world sources. Typical results are presented in figures 7 through 9. The repetitive grid structure extracted is graphically displayed overlaid upon the original image. These results clearly indicate the major result of this paper: that real-world regular repetitive textures can be analyzed by local analysis algorithms based on the dominant feature assumption and the properties of fundamental frequency vectors. Figure 7 illustrates the successful processing of a texture that is severely perspectively distorted. Our system is able to identify this texture as regular repetition because the perspective distortion does not destroy the local repetitive nature of the texture. Figure 7 also illustrates an effect of extreme texture distortion. In the lower half of the figure, the algorithm has chosen two alternative, somewhat counter-intuitive interpretations of the fundamental texture frequency. These arise because the distortion of the texture is so severe that in these portions of the image, the texture has passed the point of ambiguity and the fundamental frequency vectors have, in fact, changed.

Our system performs region segmentation on regular repetitive textures purely as a side-effect of the detailed analysis. The results are surprisingly good, as can be seen in both figures 7 and 9. In addition, the structural description produced by our system is very detailed. It may therefore be useful as a basis for a detailed shape-from-texture analysis. For example, the one-eyed stereo approach of [13] may be directly applicable to the structural grids extracted by our system.

13 Conclusion

We have made an assumption, called the dominant feature assumption, that the texels of a regular repetitive texture each contain a single feature that is more prominent than any other feature in the texture. We have also defined the fundamental frequency vectors of a regular repetitive texture as the shortest pair of frequency vectors in the texture. By identifying the dominant features and extracting their fundamental frequency structure, we have successfully extracted the structure of regular repetitive textures in real-world images. The system used for this analysis performs entirely local computations, enabling it to interpret severely distorted regular repetitive texture. In addition, segmentation of regular textures is achieved as a side-effect of their analysis.

1. References

1.) N.Ahuja and D. Haken, "Towards Integrated Vision: Representation and Three-Dimensional Interpretation of Image Texture," Technica l Report, Coordinated Science Laboratory, University of Illinois at Urbana-Champaign, July 1986.
2. D. Chetverikov, "Measuring the Degree of Texture Regularity," in *Proc. 7th Int. Joint Conf. on Pattern Recognition*, pp. 80-82, Montreal, Canada, July 1984.
3. R. Conners and C. Harlow, "Toward a Structural Analysis Based on Statistical Methods," *Computer Graphics and Image Processing* 12, pp. 224-256, 1980.
4. L. Davis, S. Johns and J. Aggarwal, "Texture Analysis using Generalized Co-occurrence Matrices," *IEEE Trans. Patt. Anal. and Mach. Intell.* PAMI-1, pp. 251-259, 1979.
5. L. Hamey, "Computer Perception of Repetitive Textures," Ph.D. thesis, Department of Computer Science, Carnegie Mellon University, Pittsburgh PA, 1988.
6. G. Hardy and E. Wright, *An Introduction to the Theory of Numbers*, Clarendon Press, 1979, chapters 3 and 24.
7. R. Kannan, "Algorithmic Geometry of Numbers," *Annual Review of Computer Science* 2, pp. 231-267, 1987.
8. J. Leu and W. Wee, "Detecting the Spatial Structure of Natural Textures Based on Shape Analysis," *Computer Vision, Graphics and Image Processing* 31, pp. 67-68, 1985.
9. T. Matsuyama, S. Miuri, and M. Nagao, "A Structural Analysis of Natural Textures by Fourier Transformation," in *Proc. 6th Int. Conf. on Pattern Recognition*, pp. 289-292, Munich, Germany, Oct. 1982.
10. T. Matsuyama, K. Saburi, and M. Nagao, "A Structural Analyzer for Regularly Arranged Textures," *Computer Graphics and Image Processing* 18, pp. 259-278, 1982.
11. R. Nevatia, K. Price, and F. Vilnrotter, "Describing Natural Textures," in *Proc. 6th Int. Joint Conf. on Art. Intell.*, pp. 642-644, 1979.
12. T. Rangarajan, "The Relative Neighbourhood Graph and the Visual Perception of Dot Patterns," Technical Report, School of Inf. and Comp. Science, Georgia Inst. of Tech., Atlanta, GA, 1982.
13. T. Strat and M. Fischler, "One-eyed Stereo: a General Approach to Modeling 3-D Scene Geometry," in *Proceedings: Image Understanding Workshop*, pp. 363-372, Miami Beach, FL, Dec. 1985.
14. F. Tomita, Y. Shirai, and S. Tsuji, "Description of Textures by a Structural Analysis," *IEEE Trans. Patt. Anal. and Mach. Intell.* PAMI-4, pp. 183-191, 1982.
15. F. Tomita, Y. Shirai, and S. Tsuji, "Description of Textures by a Structural Analysis," in *Proc. Int. Joint Conf. Artif. Intell.*, pp. 884-889, 1979.
16. G. Toussaint, "The Relative Neighbourhood Graph of a Finite Planar Set," *Pattern Recognition* 12, pp. 261-268, 1980.
17. S. Zucker and D. Terzopoulos, "Finding Structure in Co-occurrence Matrices for Texture Analysis," *Computer Graphics and Image Processing* 12, pp. 286-308, 1980.

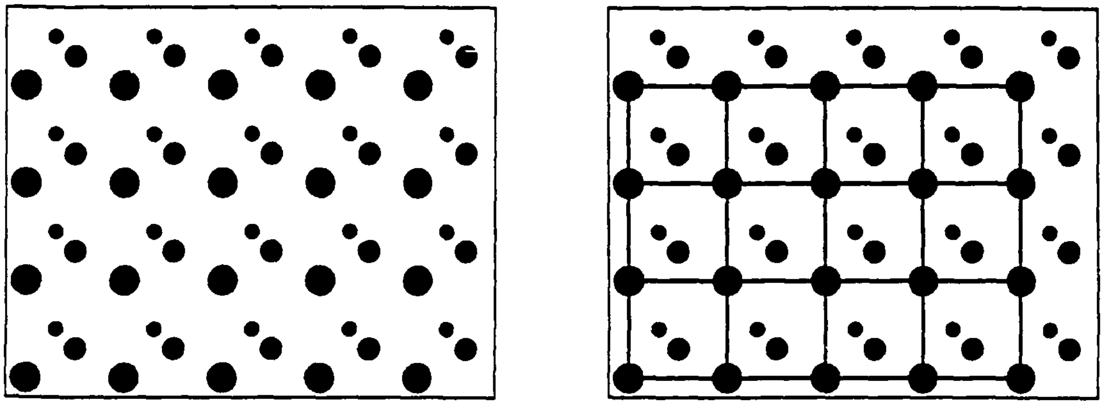


Figure 1: *Repetitive structure of dominant features of a texture.*

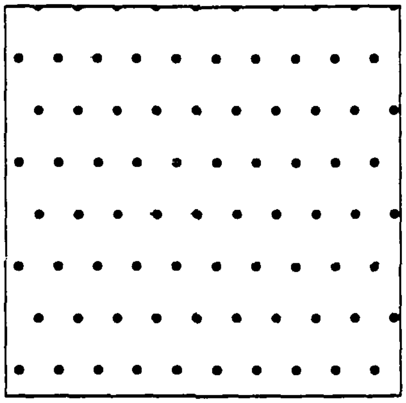


Figure 2: *An ambiguous repetitive texture.*

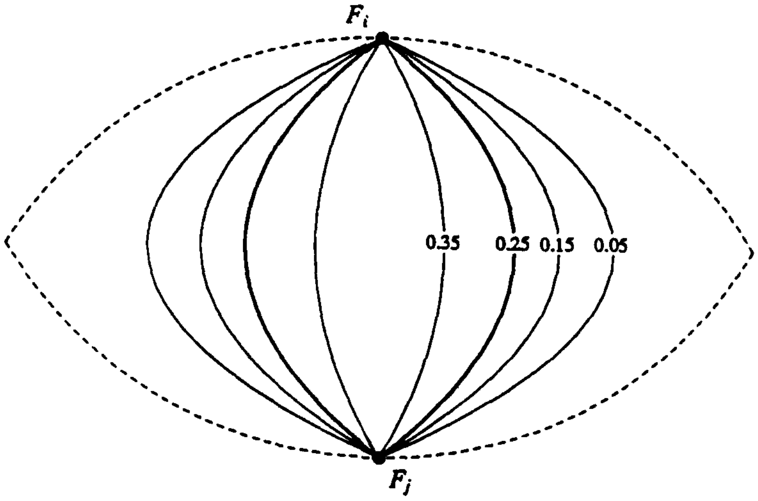
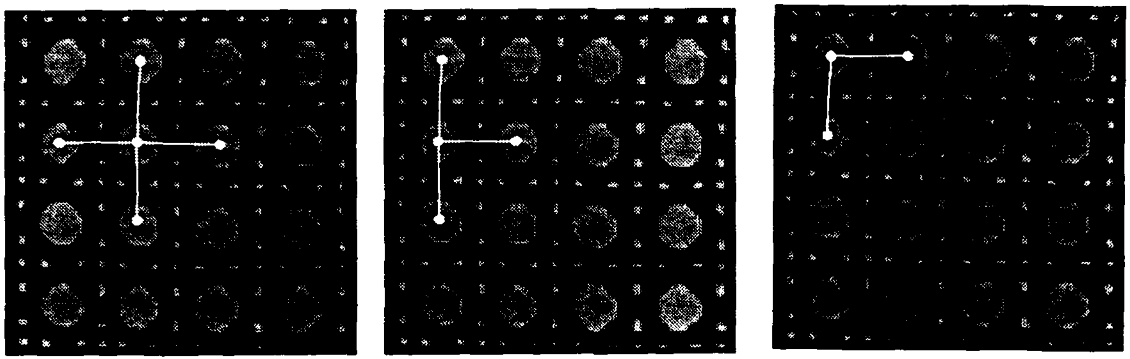


Figure 3: *Contour plot of S_L and the "lune" of F_i and F_j.*



(a) Cross relationship

(b) T-shaped relationship

(c) L-shaped relationship

Figure 4: *Types of repetitive relationships*

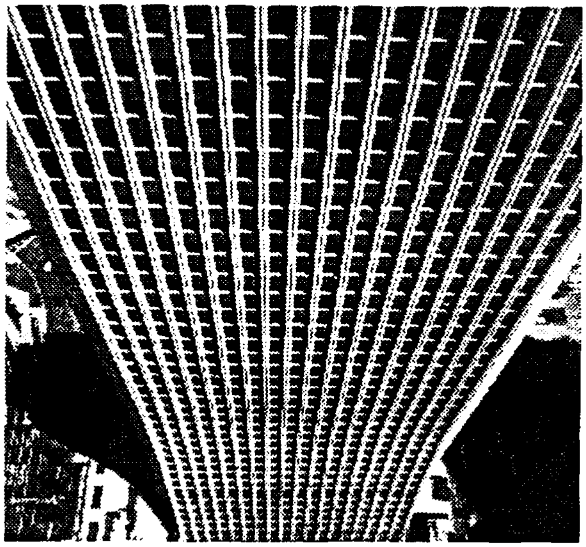


Figure 5: *Image of a skyscraper.*

Figure 6: *Features extracted from figure 5.*

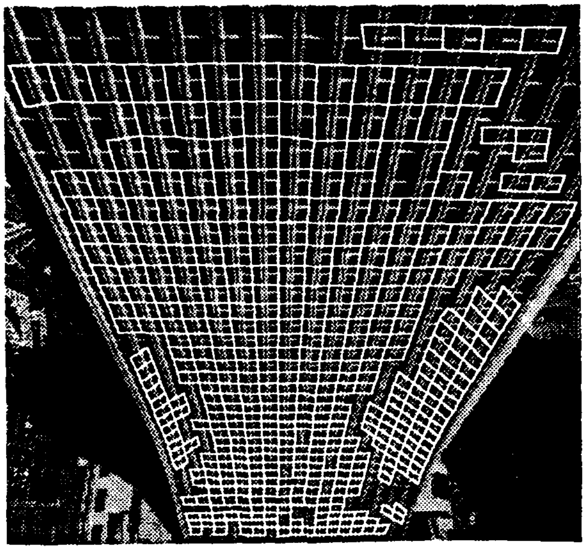


Figure 7 Caption: *Extracted repetitive structure of figure 5.*

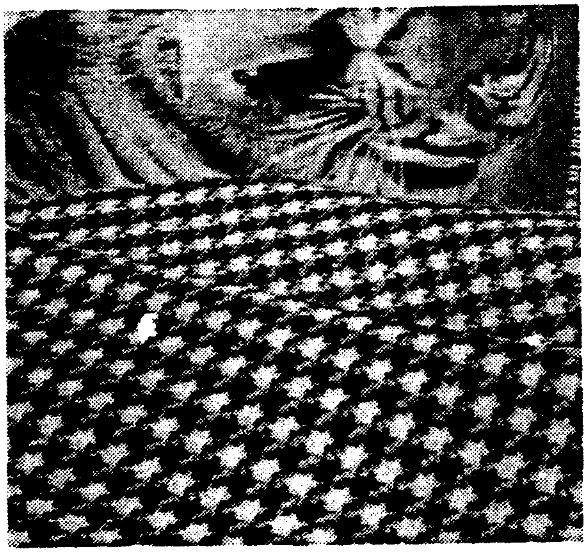


Figure 8: *Arrangement of multiple textiles and non-textiles.*

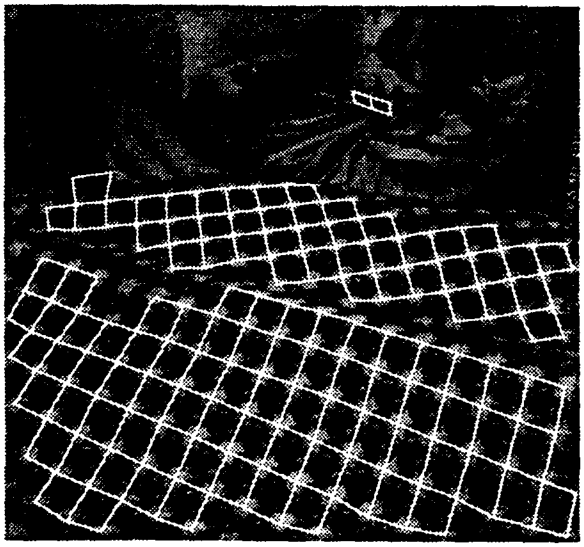


Figure 9: *Extracted repetitive structure of figure 8.*

AN APPLICATION OF DYNAMICAL SYSTEMS THEORY TO SHAPE FROM SHADING

Bror V. H. Saxberg
MIT A.I. Laboratory
NE43-792
545 Tcchnology Square
Cambridge, MA 02139

ABSTRACT

The goal of the research reported here is to gain a better theoretical understanding of what lies behind the visual system's ability to generate robust surface interpretations from single grey scale images. We have examined the image irradiance equation using notation and concepts from modern differential geometry and global analysis. The method of characteristic strips used by Horn (Horn, 1975) defines a dynamical system on a four dimensional space. Using modern methods for analyzing the behavior of dynamical systems, new uniqueness results and a new shape from shading algorithm emerge based on the image dynamical system. Solution surfaces for the shape from shading problem are invariant manifolds of the flow generated by the image dynamical system. The stable and unstable manifolds associated with certain critical points in the image play an important role in determining solution surfaces. A theorem about unstable manifolds (the Lambda Lemma) indicates a class of computational methods for finding stable and unstable manifolds around these critical points. We have implemented several such methods, and find them to be robust in the presence of image noise and mistakes in assumptions about the light source.

1 INTRODUCTION

One of the major challenges for appreciating how vision contributes to our knowledge of the world is to understand how it copes with the wide variety of lighting conditions, surfaces, and surface markings to provide accurate representations of the surfaces around us. Although there are a variety of sources of information available to the human visual system including stereo, color, motion, and shading, we get very clear impressions of the three-dimensional character of a scene from a single grey-level still picture, even of scenes or objects that are not recognized as previously having been viewed. This suggests that there is sufficient information in monocular grey-level images without motion for the visual system to arrive at a three-dimensional interpretation that is very convincing. The visual system is not always correct or even unambiguous in its interpretations: a picture can be interpreted "correctly" both as a flat surface with shading variations or as a clear window onto a scene.

It is not known how accurately the visual system estimates shape considered as the exact location in space of each point on a surface or as the exact orientation of the surface at each point. It is difficult to create psychophysical experiments to test this because it is difficult to quantitatively probe a subject's internal information about surface shape. There have been a few experiments along these lines recently (Mingolla and Todd, 1986, Bülthoff and Mallot, 1987) which suggest that our impressions of surface orientation are far more qualitative than we might like to believe. Nonetheless, we have examined how much detailed information about shape is theoretically contained in an image of a surface; answers to questions like these at least provide upper bounds on what the visual system can do in the absence of other information or assumptions.

2 BACKGROUND

Without any assumptions about lighting conditions or surface properties, there is no hope for recovering any information about surfaces in space: one can take a nearly arbitrary smooth surface and paint it to give the same image as another smooth surface without paint. The human visual system is apparently capable of using more than one set of assumptions: consider again the dichotomy between a picture as shaded surface and as window onto a scene. At the same time, it is difficult for us to entertain a continuum of possible interpretations: without extra cues, it is hard for us to convincingly interpret a flat picture of a scene as a different, curved, carefully painted surface.

Berthold Horn's early work on the shape from shading problem (Horn, 1975) examined it as a problem of physics, looking at the process of image formation and how light is reflected from objects and concentrated to form images. He defined a summary function, the reflectance function, that contained all the relevant local information about lighting conditions and surface reflecting properties under the assumption that reflecting properties of a surface patch were dependent solely on the orientation of the surface, and were constant with rotations of the surface around its normal. With additional assumptions of no cast shadows and no mutual illuminations, the brightness of a point in an image depends on the location of the point in space, the orientation of the surface in space at that point, and the reflectance function for the lighting conditions and surface material.

Assuming a known reflectance function, Horn was able to characterize the shape from shading problem as a nonlinear first order partial differential equation, the image irradiance equation:

$$E(x,y) = R(p,q),$$

using the standard (p,q) gradient space coordinate system to define orientations, and assuming orthographic projection down the z axis onto the (x,y) plane for image formation (Horn, 1975). For simplicity we assume the reflectance function acts on orientations but is independent of space coordinates; for example, a scene lit by a point source from a considerable distance.

Classically, such an equation is solved by the method of characteristics, and Horn used this technique to develop a method for solving for the surface given some initial curve lying on the surface with known surface normals. Essentially, the problem becomes a Cauchy problem; the solution proceeds along curves called characteristic strips beginning at the known curve of data. The equations that define the characteristic strips are

$$\begin{aligned} \dot{x} &= R_p \\ \dot{y} &= R_q \\ \dot{p} &= E_x \\ \dot{q} &= E_y. \end{aligned} \tag{1}$$

[There is also an equation $\dot{z} = pR_p + qR_q$; since $\dot{z}$ is a function solely of p and q, we can solve first for p, q, x, and y and then use these solutions to find $z(t)$. This allows us to consider an ordinary differential equation in just four dimensions, x, y, p, and q.] Time is a parameter along the characteristic strip $(x(t), y(t), p(t), q(t))$.

Horn recognized the importance of critical points in the image, places where image intensity as a function of image coordinates has a critical point. An isolated critical point (p_c, q_c) in the reflectance function R (i.e. a surface orientation at which $R_p = R_q = 0$) in general produces an image critical point. Given a known reflectance function, a critical point (x_c, y_c) in the image with the same brightness $E(x_c, y_c) = R(p_c, q_c)$ as at the reflectance function critical point (p_c, q_c) could be assumed to have the same surface orientation as the reflectance function critical point.

Unfortunately, as Horn noted, the characteristic trajectories cannot be used to draw out the surface from the combined critical point (x_c, y_c, p_c, q_c): $E_x = E_y = R_p = R_q = 0$ by assumption, and so the characteristic strip "emanating" from the critical point is the degenerate constant strip $(x(t), y(t), p(t), q(t)) = (x_c, y_c, p_c, q_c)$. To try to get around this difficulty, Horn constructed a spherical cap consistent with the critical point orientation and used a small closed contour on this cap as his initial condition curve.

Direct integration of the characteristic equations to find the characteristic strips suffers from noise sensitivity in practical implementations. As the solution proceeds along constructed curves from the initial condition curve, these curves can deviate as a result of quantization error and other noise influences. Horn and Brooks (Horn and Brooks, 1986) review and compare a number of related methods by different researchers for making the solution of the shape from shading problem a global one, allowing data from the full image to contribute to finding stable, relatively robust solution surfaces. They provide a recipe for generating shape from shading methods. These are relaxation methods based on minimizing a certain measure, typically the integral over the image region of some combination of the error in the image irradiance equation and a penalty term for departure from smoothness.

Various such smoothness, integrability, or regularization terms have been tried by various researchers. Horn and Brooks indicate that enforcing the correct integrability constraint, $p_y - q_x = 0$ in gradient space coordinates, does not immediately yield a convergent relaxation scheme. However, as Frankot and Chellappa (Frankot and Chellappa, 1987) point out, using a different penalty term instead may yield a non-integrable set of surface normals. In other words, the resulting set of normals may be a smoothly chosen set of unit vectors, but may not be surface normals for any possible two-dimensional surface.

Pentland takes a different approach (Pentland, 1982, 1984) from Horn. Rather than assuming full knowledge of the reflectance function, which is not available for human vision, he makes slightly less restrictive assumptions about the reflectance function (e.g. it is Lambertian, but with unspecified direction), and makes more assumptions about the surface structure. Pentland shows that if one assumes that the surface at a point is spherical, i.e. can be fitted by a spherical patch then Lambertian reflectance gives a unique solution for the surface at that point. As one might expect and as Pentland points out, not all local image intensity patterns can be accounted for by a spherical patch. Pentland also recognizes that some spherical patch solutions for certain kinds of image data are less likely to be reasonable than others.

One problem with this approach is that it is an extremely local analysis. In general, the only non-planar surfaces composed completely of equal curvature points are pieces of spheres. Although one may be able to generate a spherical solution patch locally consistent with the image for many points in an image, it is not clear that the surface normals of these patches will be able to form a smooth surface. Nor is it clear that such a surface would have much to do with the original surface imaged. To compensate for the lack of generality of this surface structure assumption, Pentland is led to statistical methods to estimate surface orientation based on correlations in natural images: heuristic estimators for surface orientation are proposed.

Frankot and Chellappa (Frankot and Chellappa, 1987) suggest a way of taking a set of non-integrable surface normals and producing an integrable set by projecting the surface normals onto the nearest set of integrable normals, where nearest is defined by a global integral distance measure defined in the gradient space coordinate system (and dependent on that coordinate system). Together with a method derived from (Horn and Brooks, 1986), they construct a more quickly convergent algorithm guaranteed to form an integral set of normal vectors, and suggest their method could be applied to Pentland's normals to generate an integrable surface.

Koenderink and van Doorn (Koenderink and van Doorn, 1980) have looked at the field of isophotes, or constant brightness contours, of an image and discussed some of their geometric features. They note that the Weingarten map, which maps the surface in space to the Gaussian sphere of unit surface normals, maps constant brightness contours on the surface to level sets of the reflectance function on the Gaussian sphere. In order to make sense of the infinity of possible arrangements of constant brightness contours potentially created by a given reflectance function, they restrict themselves to Weingarten maps that are "stable," or generic, meaning their fundamental geometry is not changed by small perturbations. This still leaves the wide class of Weingarten maps that are mostly one to one, with one-dimensional curves of points that are fold singularities of the Weingarten map, and isolated points that are cusps of the Weingarten map. They classify various regions of such a surface using parabolic lines (folds in the Weingarten map), and discuss different causes for critical points in the image: specularities, certain points on the parabolic lines, and critical points of the reflectance function itself. These latter critical points are the important critical points in both Horn's and our analysis of shape from shading.

Several difficulties and issues are common to the motivation and execution of shape from shading methods. Enforcement of integrability is one issue common to different shape from shading methods. Most of the methods struggle to ensure that the surface normals produced are from a true two-dimensional surface, recognizing that not all collections of surface normals can be considered as coming from a surface. Frankot and Chellappa (Frankot and Chellappa, 1987) make the most explicit separation between the two groups of surface normal distributions, those that are integrable and those that are not, projecting one group onto the other. In other methods, some "integrability constraint" is frequently added in as a "regularization" term that is supposed to roughly deal with getting a smooth surface as a solution. Unfortunately, solutions derived without strictly enforcing integrability may still not be integrable.

The question of uniqueness of solution for the different methods has also proved difficult. One would like some knowledge about how many interpretations of an image are possible given the information in the scene Bruss (Bruss, 1980) found a uniqueness result for Horn's method in the case of a reflectance function of a particular form: essentially those with level surfaces that are concentric ellipses in the gradient space coordinate system. Many reflectance functions do not have this form, however. Deift and Sylvester (Deift and Sylvester, 1981) investigated in some detail uniqueness results for the degenerate case of an image of a Lambertian hemisphere lit from the viewing direction. They found different classes of non-spherical, even non-symmetrical, local solutions which are C^2 almost everywhere. If the solution surface is required to be C^2 everywhere, the expected spherical solutions are unique. They used methods of functional analysis, working in a polar coordinate system to analyze this specific case, and did not address questions about stability of the unusual solutions. Brooks (Brooks, 1982) discusses the general problem of ambiguity of solutions surfaces in images, and shows families of solutions for certain degenerate cases, e.g. a plane and hemisphere lit from the viewing direction. He also briefly examines the relationship between uniqueness of solution and the kind of image patch in the case of a hemisphere: certain patches of the image provide much more constraint than others. General results on uniqueness and properties of solutions are lacking from the literature.

The stability of scene interpretation to variations in assumptions has not been fully explored. Ikeuchi and Horn (Ikeuchi and Horn, 1981) did a number of experimental tests of the performance of their shape from shading algorithm under violations of the assumed conditions, but this seems rare in the published literature.

3 DYNAMICAL SYSTEMS THEORY APPROACH

Following Horn, we assume a known reflectance function, $R(p,q)$, independent of location in space; we also assume a smooth image. We assume orthographic projection $(x,y,z) \mapsto (x,y)$ of space onto the (x,y) image plane.

A system of ordinary differential equations such as

$$\begin{aligned} \dot{x} &= R_p \\ \dot{y} &= R_q \\ \dot{p} &= E_x \\ \dot{q} &= E_y. \end{aligned} \tag{1}$$

defines a *dynamical system* (Abraham, Marsden, and Ratiu, 1983). A dynamical system can be though' of as a *vector field*, the choice of a velocity vector at each point of a space ((x,y,p,q) space here). A particle traveling through space with velocity equal to the vector field at each point of its travel is said to be moving along the *flow* of the dynamical system, or along a *trajectory* of the dynamical system. The characteristic strips are the (characteristic) trajectories of the image dynamical sytem defined by equations (1).[1]

A smooth solution surface for the shape from shading problem is a two dimensional surface in $\mathbb{R}^3$ that could have generated the image. As noted by Horn, such a surface must be made up of characteristic strips. This means that if a point (x_0, y_0, z_0) on the surface has surface orientation (p_0, q_0), then the entire characteristic strip $(x(t), y(t), z(t), p(t), q(t))$ through $(x_0, y_0, z_0, p_0, q_0)$ lies on the surface, i.e. $(x(t), y(t), z(t))$ is a curve on the surface, and $(p(t), q(t))$ gives the orientation of the solution surface at each point on the curve. Not all choices of (x_0, y_0, p_0, q_0) can be consistent with the image: we must have

$$H(x_0, y_0, p_0, q_0) \overset{\Delta}{=} E(x_0, y_0) - R(p_0, q_0) = 0$$

to begin with; the characteristic strip is then guaranteed to keep

$$H(x(t), y(t), p(t), q(t)) = 0.$$

As suggested by the above reasoning, instead of thinking of the solution surface as a two-dimensional surface sitting in $\mathbb{R}^3$, we can drop the depth coordinate (essentially depth is a symmetry in our problem: neither the image nor the reflectance function depend on it) and consider the solution surface to be a two-dimensional surface in the four dimensional space defined by the coordinates (x,y,p,q).[2] The two-dimensional solution surface is now made up of one-dimensional trajectories $(x(t), y(t), p(t), q(t))$ of the image dynamical system.

In the theory of dynamical systems, a surface that is entirely made up of trajectories is called an *invariant* surface: if a point on an invariant surface is allowed to flow along the trajectory defined by the dynamica! system, the point stays on the surface. All possible smooth solution surfaces for the image irradiance equation are invariant surfaces for the image dynamical system. The reverse turns out to be not quite true: not all invariant surfaces of the image dynamical system correspond to physical solution surfaces of the image irradiance equation. We might have an invariant surface on which $H(x,y,p,q) \neq 0$, or the invariant surface might not project into $\mathbb{R}^3$ as a two-dimensional physical surface. Nevertheless, the problem of finding solution surfaces to the image irradiance equation is closely connected to the problem of finding two-dimensional invariant manifolds of the image dynamical system.

[1] The image dynamical system can be shown to exist independent of a particular coordinate system choice using differential forms and the Frobenius theorem (Saxberg 1989); this can be useful if one wishes to examine the dynamical system with an unusual coordinate system, e.g. examining the shape from shading problem along the bounding contour (where the image projection just grazes the surface) using a coordinate system that follows the bounding contour and gives the shape a particularly simple form (Saxberg, 1989).

[2] Technically we can only do this globally if we work in a coordinate independent fashion (Saxberg, 1989); practically, with the (x,y,p,q) coordinate system we are only excluding portions of solution surfaces that are on the bounding contour.

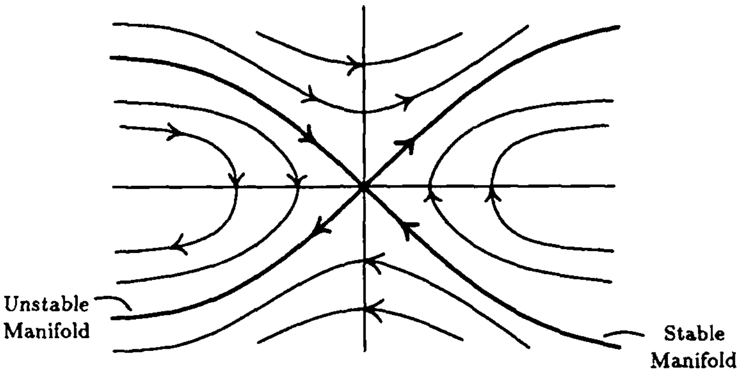


Figure 1. Two-dimensional dynamical system with saddle critical point.

3.1 THEORETICAL RESULTS

In the theory of dynamical systems, the study of critical points has been found to be very useful in characterizing the behavior of a dynamical system. A critical point p_c in a dynamical system $\dot{p} = f(p)$ is a point where the associated velocity vector is the zero vector, $f(p_c) = 0$. In the case of the image dynamical system defined by equations (1), this occurs when

$$R_p = R_q = E_x = E_y = 0,$$

i.e. at critical points in the image due to critical points in the reflectance function. Near a critical point of a dynamical system $\dot{p} = f(p)$, one can gain insight into the behavior of the system by studying the linearization $\dot{p} = Ap \overset{\Delta}{=} [\partial f_i/\partial x_j]\,p$ of the system around the critical point; the matrix A is essentially the Jacobian of the vector field f at the critical point. The eigenvectors and eigenvalues of A give qualitative information about the behavior of the non-linear dynamical system near the critical point (Abraham, Marsden, and Ratiu, 1983).

In the case where none of the eigenvalues have real part equal to zero, the critical point is called hyperbolic. In this case, the trajectories around the critical point in the linearized system are homeomorphic to the trajectories in the non-linear system (the Grobman-Hartman Theorem, (Palis and de Melo, 1982)): the linear and nonlinear systems look alike (aside from some bending) in some neighborhood of the critical point. Existence and uniqueness of invariant manifolds for the linearized system imply the existence and uniqueness of invariant manifolds of the non-linear system near the critical point.

Hyperbolic critical points are generic; i.e., unless there are special symmetries operating, the "usual" type of critical point is a hyperbolic one, and slight perturbations of a dynamical system with a hyperbolic critical point have a similar hyperbolic critical point near the original one. It can be shown (Saxberg, 1989) that critical points of an image due to an isolated generic maximum (or minimum) of a reflectance function give rise almost always to hyperbolic critical points of the image dynamical system for a generic surface. We call these hyperbolic critical points of the image dynamical system *very good* critical points. Non-hyperbolic critical points may occur if the critical point happens to fall on a point of zero Gaussian curvature, e.g., a parabolic line.

Very good critical points turn out to have four real eigenvalues, $\pm\lambda_1$, $\pm\lambda_2$ (Saxberg, 1989). A two-dimensional analogue is shown in Figure 1: here, there are two eigenvalues, one positive and one negative. Because there are both positive and negative eigenvalues, the dynamical system has a complicated saddle-type set of trajectories near it. The eigenvectors are parallel to the bold diagonal trajectories at the origin. Note that these trajectories are the only ones that actually approach the critical point; the other trajectories only get within a finite distance from the critical point before moving away again. These two diagonal curves are the only one-dimensional invariant manifolds containing the critical point.

In the shape from shading case, we are interested in two-dimensional invariant manifolds containing the critical point. Within the set of such manifolds we will find all smooth solution surfaces for the shape from shading problem that are consistent with the critical point. Because we have both positive and negative eigenvalues at a very good critical point, most of the one-dimensional trajectories making up invariant manifolds in the four dimensional (x, y, p, q) space will not actually approach the critical point; thus, there will not be many invariant manifolds which contain the critical point.

We can use the linearization of the image dynamical system to tell us how many there are likely to be, since invariant manifolds for the linearized system correspond to invariant manifolds of the non-linear system on a neighborhood of the critical point. Pairs of eigenvectors span two-dimensional invariant subspaces for the linearized system, and in fact these subspaces are tangent to the non-linear invariant manifolds at the critical point. In the generic case, there will be four distinct eigenvectors; of the six possible pairs of eigenvectors, two can be shown to be non-physical (Saxberg, 1989), leaving four potential candidates for solution surfaces.

Two of these are the stable and unstable manifold for the critical point. The stable subspace is spanned by all eigenvectors with negative eigenvalues; the unstable subspace is spanned by all eigenvectors with positive eigenvalues. The stable and unstable manifolds contain all those trajectores that head towards the critical point with increasing time, and all those trajectories that head towards the critical point with decreasing time (head "away" from the critical point with increasing time) respectively. A major theorem of dynamical systems theory, the Stable Manifold Theorem (Abraham and Marsden, 1985), says that such manifolds exist, are smooth (for a smooth dynamical system), and are unique for any critical point (not just hyperbolic critical points). It turns out (Saxberg, 1989) that at a very good critical point due to a maximum in the reflectance function, the unstable manifold corresponds to a concave solution surface, while the stable manifold corresponds to a convex solution surface. The image dynamical system restricted to these invariant manifolds looks like a sink or source.

There may be two other potential solution surfaces. These are determined by the invariant subspaces of the linearized system spanned by one eigenvector with positive eigenvalue and one eigenvector with negative eigenvalue. These correspond to saddle-shaped surfaces in space. The image dynamical system restricted to these invariant manifolds looks like a saddle system (as in Figure 1). This also means only two trajectories on the invariant surface actually approach the critical point; only these two trajectories in general are forced to have $H(x, y, p, q) = 0$; the other trajectories making up the invariant surface may have other values for H. Thus, these saddle invariant surfaces may not always represent correct solutions to the shape from shading problem, which must have $H = 0$ over the entire surface. However, if a very good critical point in fact occurs at a saddle-shaped part of a physical surface, the invariant manifold associated with it is of this type.

In summary, in some neighborhood of a generic critical point of an image of a generic (unpainted) surface due to a known local (generic) maximum of the reflectance function, there are at least two and at most four possible smooth solution surfaces for the image irradiance equation: one convex, one concave, and possibly two that are saddle-shaped. Cases where this result may not hold are unstable, in that arbitrary small perturbations of the surface, or the reflectance function, brings one back to a situation where the result holds. Such cases include zero Gaussian curvature of the surface at the critical point, or unusual flatness of the reflectance function (i.e. zero second derivatives in some direction at the critical point), or rotational symmetry of the image and the reflectance function around the projection direction. In this latter case, the stable and unstable manifolds are still unique, but there are an infinite number of saddle invariant manifolds related by rotation; thus, there are unique convex and concave solutions, but potentially an infinite family of saddle solutions.

3.1.1 The Fundamental Instability of a True Image Irradiance Equation

A true solution surface for an entire image should consist of a single two-dimensional invariant manifold which is the unstable manifold for some critical points, the stable manifold for others, and perhaps a saddle invariant manifold for the rest. It turns out that this is very unusual behavior for a dynamical system. Generically, two two-dimensional invariant manifolds for different critical points will only intersect (if they intersect) along a one-dimensional curve (Abraham and Marsden, 1985).

We can gain some insight into the situation by looking again at two dimensions. In Figure 2 we show a piece of a two-dimensional dynamical system with two critical points, each of which has both a stable and an unstable manifold. In Figure 2a and 2c, these manifolds do not intersect; the unstable manifold for one critical point is just another trajectory for the other critical point. The case where they intersect is shown in Figure 2b. If Figure 2b is perturbed generically, it will fall apart to be either like Figure 2a or 2c.

The image dynamical system due to a real surface is therefore a very delicate thing seen from a generic perspective. This has two consequences, one bad, one potentially good. The bad news is that as we numerically try to find, say, the unstable manifold of a critical point (by directly drawing out trajectories or by the methods in the next section), errors will occur as if we had randomly perturbed the dynamical system. It is very unlikely that the perturbed unstable manifold will merge with the perturbed invariant manifold coming from another critical point and create a single, smooth, two-dimensional surface. The good news is that that this may provide a way to tell a bad reflectance function choice from a good one: if the invariant manifolds do not "nearly" match up, we must be on the wrong track. Either the reflectance function is bad, or there is no surface corresponding to the image.

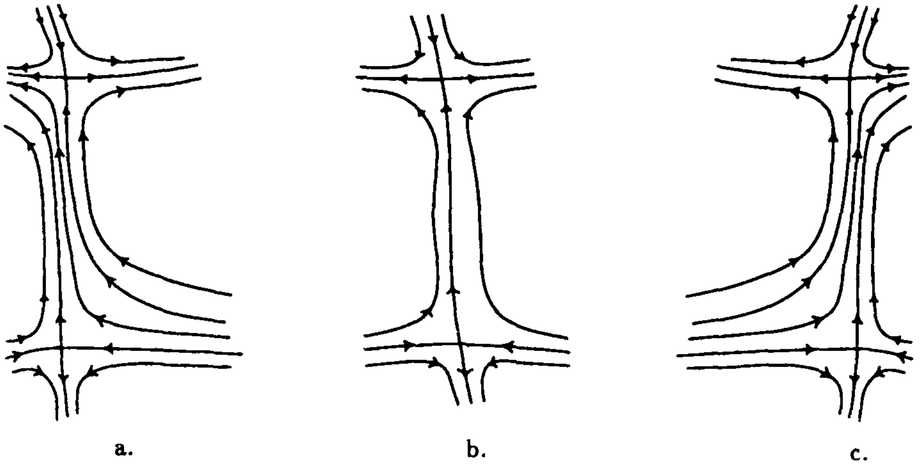


Figure 2. Two critical points of a two dimensional dynamical system: a. Non-intersecting stable and unstable manifolds. b. Intersecting stable and unstable manifolds. c. Non-intersecting stable and unstable manifolds.

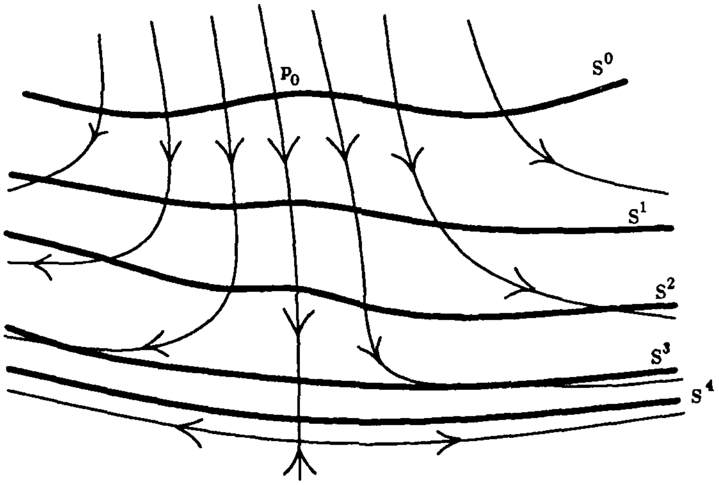


Figure 3. Initial manifold is S^0. p_0 is at the intersection of S^0 with the stable manifold. The S^i are the deformations of S^0 by the flow of the dynamical system.

Catastrophe theory results suggest that if the true reflectance function comes from a parameterized set of reflectance functions, it may be possible to adjust the parameters to find the correct invariant manifold solution and the correct reflectance function. If so, this adjustment process may work (i.e. may be stable) even under perturbations of the image and reflectance function.

3.2 ALGORITHMIC RESULTS

As a result of the theoretical results of the last section, we are quite interested in methods for finding the stable and unstable manifolds of an image dynamical system near a critical point, as these will give the unique convex and concave solution surfaces consistent with the image and known reflectance function. Another theorem from dynamical systems, the Lambda Lemma (Palis and de Melo, 1982), suggests a class of methods to do this.

We start with an inital two-dimensional manifold, S^0, which intersects the stable manifold at an angle (transversely) (Figure 3); because the stable manifold and the initial manifold are both two-dimensional, in the four-dimensional space they will have a point p_0 as transversal intersection. If we allow the initial manifold to be deformed by the image dynamical flow (letting each point on S^0 be transported by the trajectory through it over time), then the point p_0 will flow towards the critical point. The Lambda Lemma says that the rest of the surface will be deformed in the limit to approach the unstable manifold, and the approach will be C^1, i.e., the tangents to the deforming surface will also approach the tangents of the unstable manifold. As p_0 approaches the critical point, the surface gets stretched out greatly from p_0, and points on the surface near p_0 are driven towards the unstable manifold.

This suggests finding methods to take an initial two-dimensional surface in (x, y, p, q) space and deform it over time using the image dynamical flow defined by equations (1). The deformed surface should converge to the unstable manifold. If we want to find the stable manifold, we can simply reverse time.

One way to do this involves fixing a grid of points on the (x, y) plane, and examining what happens to a two-dimensional surface defined as $(x, y, p(x, y), q(x, y))$ when it is deformed by the image dynamical flow. It turns out (Saxberg, 1989) that we can examine analytically what happens to $p(x, y)$ and $q(x, y)$ directly: for some small δt, we get

$$\begin{aligned} p_{new}(x, y) &\approx p(x, y) + \delta t \cdot v_p \\ q_{new}(x, y) &\approx q(x, y) + \delta t \cdot v_q, \end{aligned} \tag{3}$$

where

$$\begin{aligned} v_p &= E_x - p_x R_p - p_y R_q \\ v_q &= E_y - q_x R_p - q_y R_q. \end{aligned} \tag{4}$$

In other words, flowing for a short period δt is approximately the same as changing the height functions $p(x, y)$ and $q(x, y)$ by $\delta t \cdot v_p$ and $\delta t \cdot v_q$ respectively.

3.2.1 Experiments

We have implemented this as a discrete iterative method on a Connection Machine. The Connection Machine is a highly parallel computer consisting of thousands of simple processors and a very fast disk drive system which can emulate a grid of processors. We used a 16k CM-1, which has 4k of memory for each of 16k processors. The kind of algorithms we have found are well suited to this kind of architecture with a SIMD language called *LISP as interface, since each processor can be assigned to a pixel of the image and operates in a neighborhood of that pixel in parallel with all the other processors.

We configured the processors as a 128 × 128 grid. For the fixed grid algorithm, each processor can be thought of as stuck to an unmoving (x, y) coordinate grid (the integer coordinates) parallel to the image plane. At each processor we can store values as if in an intelligent array: we keep values for the image gradients E_x and E_y as well as current estimates of the deformed surface heights p and q for each (x, y). We use an initial flat surface given by $p(x, y) = q(x, y) = 0$.

We need to compute the "surface vector field" components v_p and v_q at each iteration:

$$\begin{aligned} v_p &= E_x - p_x R_p - p_y R_q \\ v_q &= E_y - q_x R_p - q_y R_q, \end{aligned}$$

where p_x, p_y, q_x, and q_y are derivatives of the current p and q iterates.

For both image intensities and p and q values, we use a simple first order method to compute the derivatives on the interior of the 128×128 square of processors at integer (x, y) coordinates: for example, if f is the function for which we want a partial derivative, we take $f_x(x, y) \approx (1/2)(f(x+1, y) - f(x-1, y))$. At the four boundaries of the square grid, we cannot use this; we use the unbalanced estimates given by subtracting nearest neighbors where we have to. For example, at the left edge we take $f_x(0, y) \approx f(1, y) - f(0, y)$; at the top we take $f_y(x, 0) \approx f(x, 1) - f(x, 0)$; at the corners of the grid, both f_x and f_y are approximated this way. We use this to estimate p_x, p_y, q_x, q_y, E_x, and E_y.

For the reflectance derivatives, we use an analytic model to give us exact values for the derivatives. In our case, we assume a Lambertian reflectance function,

$$R(p, q) = \alpha \frac{p l_1 + q l_2 - l_3}{\sqrt{p^2 + q^2 + 1}\sqrt{l_1^2 + l_2^2 + l_3^2}}$$

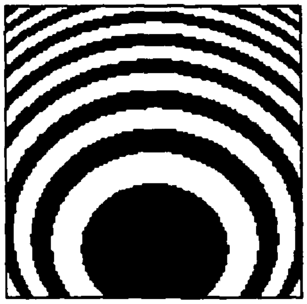


Figure 4. Constant brightness contours for sphere. Each stripe is 15 grey levels wide.

where (l_1, l_2, l_3) gives the orientation of the light source and α is an albedo factor; for our purposes, we take $\alpha = 1.0$. From this, we have

$$R_p(p,q) = \frac{\alpha}{\sqrt{l_1^2 + l_2^2 + l_3^2}} \frac{l_1 q^2 + l_1 - (q l_2 - l_3) p}{(p^2 + q^2 + 1)^{3/2}}$$

$$R_q(p,q) = \frac{\alpha}{\sqrt{l_1^2 + l_2^2 + l_3^2}} \frac{l_2 p^2 + l_2 - (p l_1 - l_3) q}{(p^2 + q^2 + 1)^{3/2}},$$

and we compute R_p and R_q at each processor using the current values of p and q.

To approximately deform the surface to match the dynamical system flow, we update the p and q values at each processor to p_{new} and q_{new} as follows:

$$p_{new} = p + h v_p$$

$$q_{new} = q + h v_q,$$

where h acts like an integration step size if we consider the procedure as a parallel integration of a vector field. In the simple examples with which we have worked, we have left h constant over all processors.

If we run just this algorithm we find that it does not converge. We have found it necessary to do a small amount of smoothing of the values of p and q to avoid "checkerboard" instabilities: small amounts of noise can explode into large alternating sections of values if the p's and q's are not smoothed between iterations. This was also found by Ikeuchi and Horn (Ikeuchi and Horn, 1981), and we find good results by performing the simple averaging operation

$$f_{new}(x,y) = (f(x+1,y) + f(x-1,y) + f(x,y+1) + f(x,y-1))/4$$

twice in succession on the p and q arrays.

This internal smoothing operation, although numerically useful, does slightly distort the final solution away from the true invariant manifold. Using an image of a sphere with radius 100 and light source located in the direction $(0, .5, -1)$, with one internal smoothing operation, the errors in the converged p and q are less than 5% almost everywhere, and less than 2% in the image interior. Even with two internal smoothing operations, the errors are less than 5% through almost all the image—where true p and q values are very near 0, of course, the relative errors are higher, and right at the borders of the image the relative errors are also higher (but never more than about 15%). We use the double smoothing because of the noise immunity it seems to give.

In Figure 4, we show the constant brightness contours for the noise-free 128 × 128 simulated image of a sphere with radius 100, Lambertian reflectance function, and light source located at $(0, .5, -1)$. In Figure 5 we show images from the approximate solutions as the iterations proceed (step size $h = 1.0$): after varying numbers of iterations, we use the current p and q values and the reflectance function to generate an image. As one can see, the images generated by the p and q estimates do show convergence to the original image through almost all of the interior of the image as expected. We have also generated images representing the errors in the p and q values: we take $255 * 10 * |p - p_{\text{true}}|$ and $255 * 10 * |q - q_{\text{true}}|$ and treat these as grey levels for an error image. A region that is just barely completely white (a grey level just reaching 255) would represent errors of .10 in p or q, corresponding to an error of about 6° in the surface normal at $p_{\text{true}} = 0$ $q_{\text{true}} = 0$, and less as p and q increase.

If we add noise to the original image, we can degrade the performance of the algorithm. With the reflectance function we are using, the maximum brightness of the image is 1.0. If we add uncorrelated uniformly distributed noise to the image with maximum value .02 (meaning the noise is uniformly distributed between ±.02), we usually still get convergence; if we add noise with maximum value .12, we do not get convergence to a possible solution. Figure 6 shows a noisy image (via constant brightness contours) with .02 noise maximum; Figure 7a shows the image

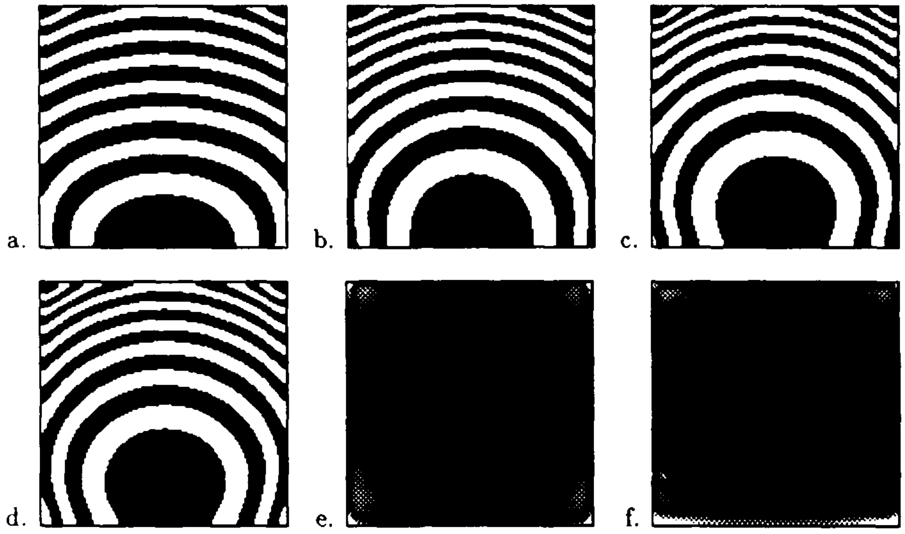


Figure 5. Iterates for fixed grid shape from shading algorithm: a. 100 iterations. b. 200 iterations. c. 400 iterations; d. 800 iterations. e. 800 iteration p error image ($255 * 10 * |p - p_{\text{true}}|$). f. 800 iteration q error image ($255 * 10 * |q - q_{\text{true}}|$).

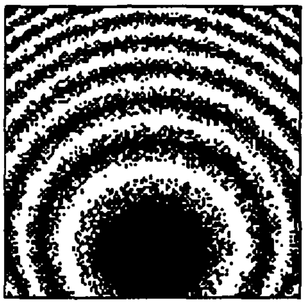


Figure 6. Noisy sphere image: Noise maximum is ±.02.

formed from the p and q arrays after 800 iterations with .02 noise, and Figure 7b and c show the p and q error images. Figure 7d shows a failed effort with noise maximum .12.

To help deal with increasing noise we can decrease the step size h and smooth the image. If we set the step size to $h = .25$, and smooth the image with one averaging operation, the algorithm converges almost always (sometimes slowly) with maximum noise values of .12; without the averaging operation, it converges only about half the time (i.e., if we run the algorithm on ten noisy examples, only five converge). With four averaging operations, the algorithm converges almost always for noise as high as .25, although quite slowly: in Figure 9 we show the contours of the image being converged to after 2400 iterations. In Figure 8 we show the original image data after adding ±.25 uniformly distributed noise and after pre-filtering; we show the image in two ways: in Figure 8a, the constant brightness intervals are the same as before, and in Figure 8b, the intervals are three times as large. The constant brightness image on the left looks very chaotic because the noise is large compared to the constant brightness intervals even after pre-filtering. To the eye, the original image itself looks fuzzy, but is clearly interpretable as a smoothly curving object.

Note that in general the algorithm does not converge to p and q values that exactly duplicate the smoothed noisy image, e.g. some complicated faceted figure whose noise-free image would be the smoothed noisy image we are working with. Instead, due to the internal smoothing operations done on the p and q arrays within the algorithm, the algorithm appears to converge to a surface that is smoother than the image data.

With as much noise as in Figure 8, the converged p and q arrays do show considerable differences from the original noise-free p and q values used to generate the images. For example, although the image in Figure 9a appears to be an adequate match for the original image, the p's and q's that generate that image vary randomly from the original image by a fair bit, as seen in the image of p and q errors in Figure 9b and c. Indeed, one would be rightly

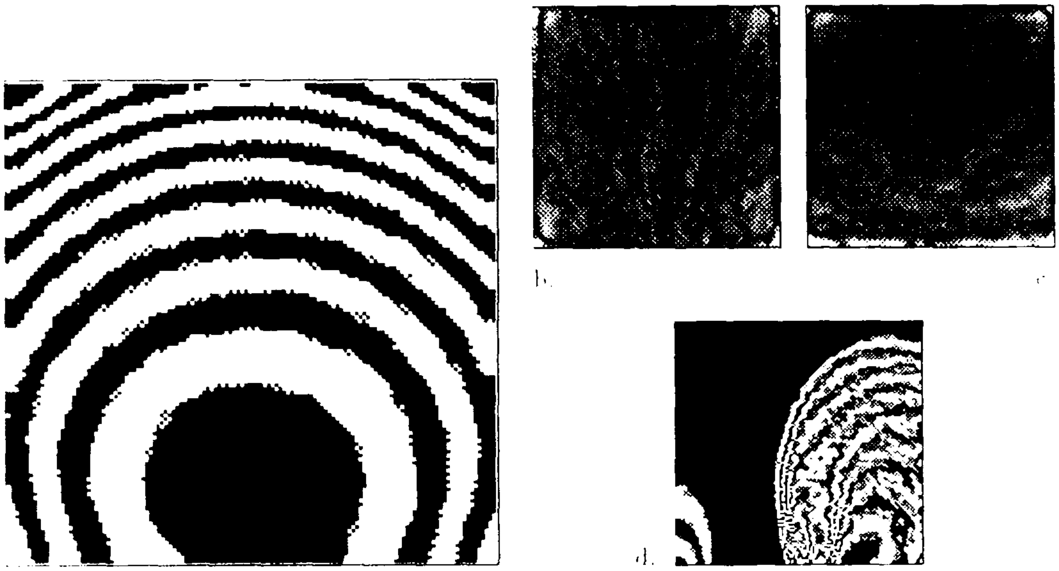


Figure 7. a. Estimated image from noisy images (±.02) after 800 iterations. b. p error image. c. q error image. d. [illegible] image for noise maximum .12 after 800 iterations.

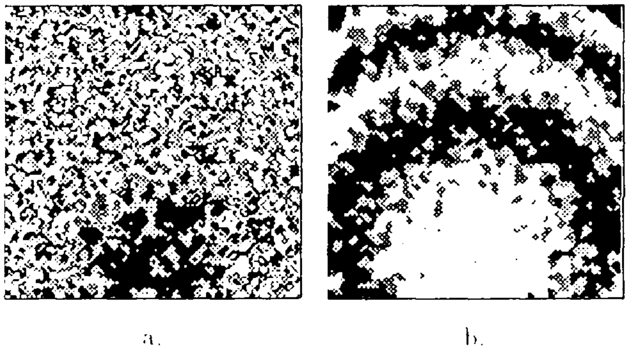


Figure 8. Noisy (±25) smoothed (four times) image: a. Using the usual constant brightness contours (every 15 grey levels). [illegible] b. Broader constant brightness contours (every 45 grey levels).

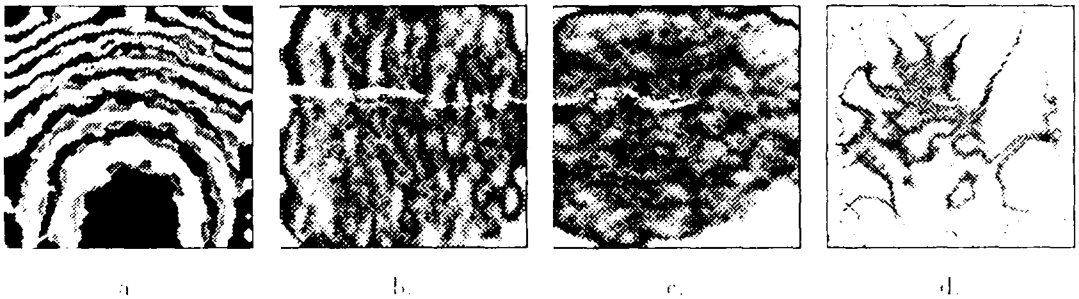


Figure 9. Noise ±25, $\lambda = 25$; smoothings: 4; iterations: 2400. a. Image from p and q values. b. p error picture. c. q error picture. d. Integrability measure picture (explained later).

[illegible] that the p and q values converged to for noisy images are not an integrable set of normal vectors. We [illegible] need some kind of constrained method like Horn and Ikeuchi's or Frankot and Chellapa's to reconstruct the

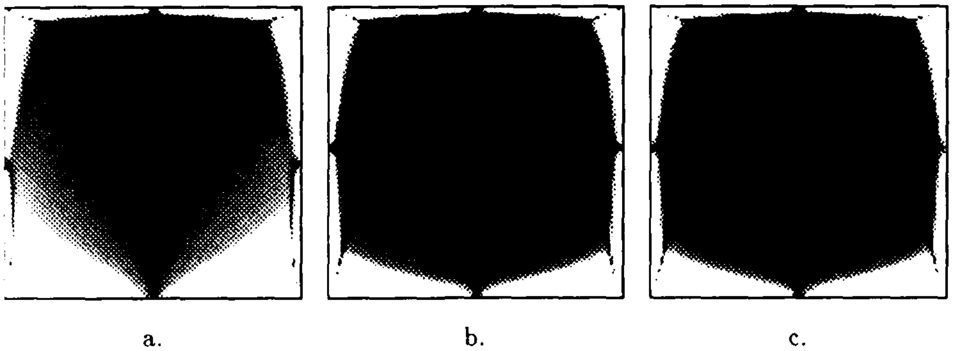


Figure 10. Integrability pictures for iterations of the noise-free image of a sphere. a. 200 iterations. b. 400 iterations. c. 800 iterations.

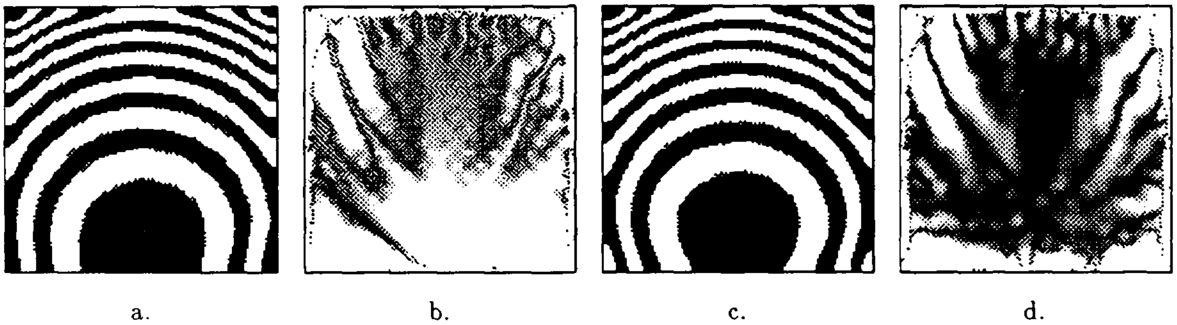


Figure 11. Integrability measures in the case of noise: noise maximum .02, h= 1.0, 4 smoothings of the image. a. 800 iterations. b. IM image after 800 iterations. c. 1600 iteratiɔns. d. IM image after 1600 iterations.

depth values from the more or less noisy p and q values. Note that the amount of noise in the p and q values is directly related to the amount of noise in the image.

In fact, one of the interesting aspects of this algorithm is the relative independence from an explicit integrability constraint to find a solution surface. If we actually converge to the theoretical unstable manifold of a smooth image dynamical system, that unstable manifold is an integrable collection of normal vectors because it is made up of characteristic trajectories. When we add noise to the image and include numerical effects, we are effectively adding noise to the dynamical system, and the unstable manifold of the new dynamical system is no longer exactly integrable (e.g. Figure 9); however, if the noise is small, the normals will be very nearly integrable (Figure 7).

An integrability measure can be used to monitor the progress of the algorithm. In order for the collection of normal vectors to be normal vectors for a real surface in space, we must have $p_y = q_x$; we can use $|p_y - q_x|$ as a measure of the integrability of the iterated surface and therefore as a measure of how close we might be to the correct unstable manifold of the image dynamical system.

Figure 10 gives a pictoral representation of this integrability measure (IM) in the noise-free case (Figure 5). The brightness values run from 0 to 255; these images are images of $10^6|p_y - q_x|$ with truncation at values greater than 255, so that the white area area represents IM values of larger than 2.5×10^{-4}. The nearly black region (represented here as having very few white dots) have $p_y - q_x$ values a small fraction ($< 1\%$) of most of the p_y values in the region. As iterations proceed, the regions of good integrability increase, although the edges of the image, where the converged constant brightness contours do not match the original (compare Figures 4 and 5), remain with relatively high IM values.

Adding noise to the image clearly degrades the integrability measure of the iterated solutions. In Figure 11 we show some examples of integrability measure pictures under noisy conditions: we look at the IM pictures for a particular image with .02 maximum uniformly distributed noise added, with step size of $h = 1.0$ and four prefilterings of the image: as the iterations proceed, the results become more and more integrable. This is consistent with the theoretical view that we are approaching the unstable manifold of the image dynamical system which should be perfectly integrable; in fact, however, progress is essentially halted after 1600 iterations: numerically, almost no

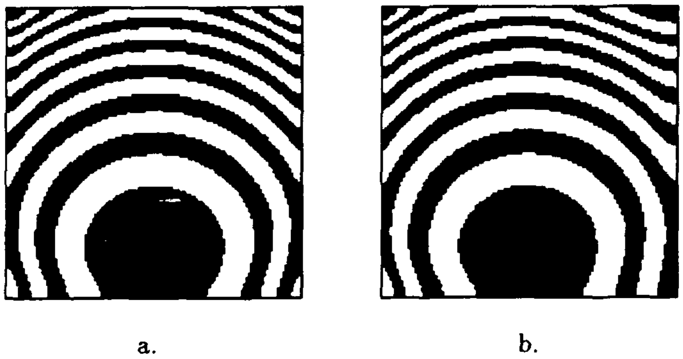


Figure 12. Incorrect reflectance function: images of p and q arrays with assumed reflectance function. Noise-free, 800 iterations, $h = 1.0$, correct $l_1 = 0.0$. a. $l_1 = .10$. b. $l_1 = .5$.

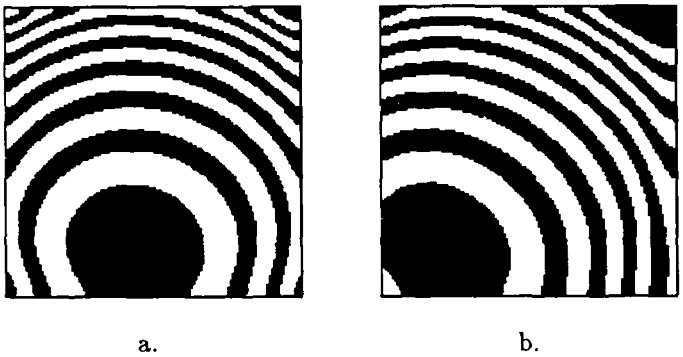


Figure 13. Incorrect reflectance function: images of p and q arrays with correct reflectance function. Noise-free, 800 iterations, $h = 1.0$, correct $l_1 = 0.0$. a. $l_1 = .10$. b. $l_1 = .5$.

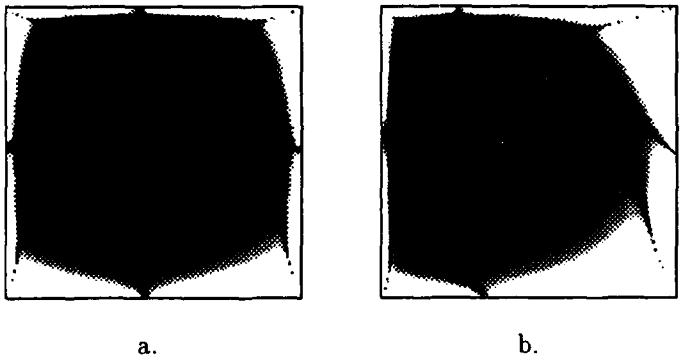


Figure 14. Incorrect reflectance function: Integrability measure pictures. Noise-free, 800 iterations, $h = 1.0$, correct $l_1 = 0.0$. a. $l_1 = .10$. b. $l_1 = .5$.

more changes in the solutions or in the IM image occur. Figure 9d shows the integrability picture for noise maximum of .25; clearly, more noise leads to worse integrability.

We can also examine the sensitivity of the algorithm to changes in the reflectance function. In Figures 12 to 14, we explore what happens if we make errors in the direction of the light source assumed to have generated the image. We assume different values for l_1, where $(l_1, l_2, l_3) = (0, .5, -1)$ is the original light source direction. As the error gets worse, the convergence of the algorithm after 800 iterations is not as complete; this can also be seen in the integrability pictures in Figure 14. Nonetheless, the arrays of p and q reached do generate images that correspond to

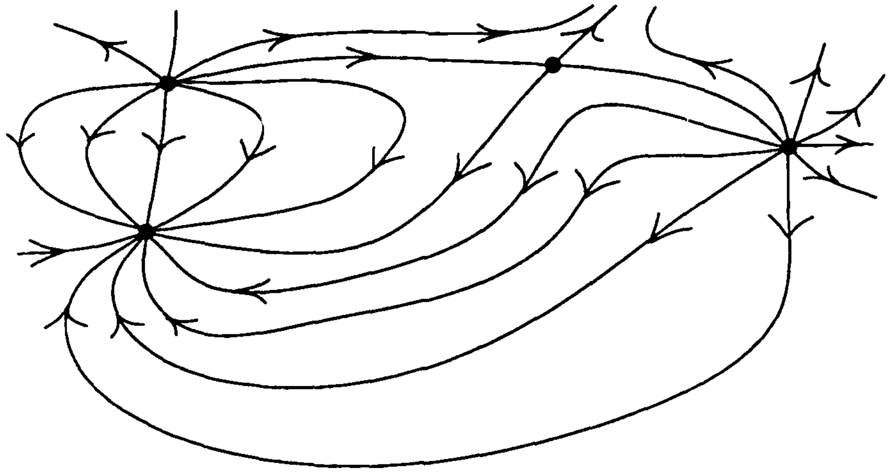


Figure 15. 2-d dynamical system restricted to the invariant manifold: in general, most points will be in the stable or unstable manifold of either a source or sink critical point.

the original over large areas with the assumed reflectance function (Figure 12); however, the surfaces themselves are increasingly different from the correct surface, as can be seen from the images generated by the converged surface and the correct reflectance function (Figure 13).

3.2.2 Conclusions on Algorithms

The fixed grid method appears to generate good solutions even in the presence of noise and seems to degrade gracefully if assumptions about the light source direction are incorrect. This is consistent with the theoretical underpinnings of the method: noise in the image or a poor choice of reflectance function represent perturbations of the original image dynamical system. Since the usual very good critical point we deal with is hyperbolic and therefore generic, perturbing the dynamical system a small amount moves the critical point a small amount and changes the invariant manifolds a small amount: the unstable manifold of the critical point is a stable feature (in this sense) of the dynamical system.

This particular implementation shares difficulties with the simple Euler method of integrating a vector field, which sequentially adds a scalar multiple of the vector field to a point on the developing trajectory to generate the next point. For example, if the step size here is too big, the method may bounce around and not converge at all. Smaller step sizes avoid this problem, but too small a step makes for slow progress; some kind of adaptive step size setting would be useful. Note that because of the smoothing of the p and q arrays as part of each iteration step, taking an exceptionally small step size effectively allows more smoothing and less deforming along the dynamical flow. There may be other methods that are analogs to the Runge–Kutta methods that make efficient use of several evaluations of the vector field to improve the estimate of the next point on the trajectory.

A feature that may be exploited is the fact that we are really interested in the characteristic paths, not the trajectories. This suggests we could use a different (positive) $h(x, y)$ to multiply the vector components v_p and v_q for each point (x, y); this would change how the initial surface is deformed, but would not change the surface that is converged to in the limit. The limit is determined by the characteristic paths and the directions they are traversed, not by how fast the paths are traversed through time; changing the magnitudes of v_p and v_q by a (positive) scalar at each point of the image should not change the theoretical limiting behavior, and may be useful in dealing with difficult regions.

Other implementations of the Lambda Lemma idea have been tried. One can place the grid of processors on the initial surface rather than on the image plane. In this case, the processors follow trajectories of the image dynamical system, and some mechanism is used to fill in the gaps as the processors draw away from each other—filling in the gaps seems to be the hard part. Another method allows each processor to control a volume of "air space" (all p and q values above a small square of (x, y) values) as an "air traffic controller": each processor controls a point on a trajectory while that point is in its airspace. New points are generated to fill holes in the deforming surface.

If an image critical point is on the stable manifold, we can reverse time and use the same techniques. If an image critical point is a saddle critical point (for example, at a maximum of the reflectance function and with saddle surface structure at that point), then the invariant manifold is neither a stable or unstable manifold for the dynamical system, and the methods described here fail. On the other hand, this saddle invariant manifold has to come from somewhere. A true solution surface will have a two-dimensional restricted image dynamical system defined on its interior. This system will have critical points, and will (probably) be a generic two-dimensional system: this means that almost all the points on the surface will either be in the stable or unstable manifold of a source or sink of the two-dimensional system (Figure 15). Almost all the points near the saddle critical point in theory should be reachable by finding the stable and unstable manifolds of other critical points.

Lambda Lemma methods will only work on some (possibly quite extended) neighborhood of a single critical point. If two critical points are contained in a neighborhood on which these algorithms are applied and the true solution surface is the stable manifold for one of the critical points and the unstable manifold for the other, it is not clear what solution (if any) will be converged to. Lambda Lemma algorithms will seek simultaneously to find the unstable manifold for both critical points; unless this happens to be a possible solution surface, the method will probably not converge.

The development of a system to actually solve shape from shading problems using these methods would be interesting. First, some reliable method of finding the stable and unstable manifolds of critical points drawn out as far as possible would be needed, perhaps including adaptive determination of the convergence region. Second, some comparison procedure to decide whether two such manifolds are the same or different would be needed: as discussed above, because of computational errors, exact matching cannot be expected even if the reflectance function is chosen exactly right. One way to accomplish the matching could be to tesselate the image into regions centered over what one hopes are very good critical points; for a typical smooth gently curved surface, there should be few such points (although in theory there could be many). At each critical point, the stable and unstable manifold could be computed; if a region contains bounding contour elements, this may be used to discard certain solutions if the invariant manifold does not come "close enough" to the bounding contour within a feasible distance from the observer. Choices from the remaining sets of invariant manifolds would have to be stitched together along the boundaries of regions: choices that were not "close enough" to each other would be discarded. There is one additional complexity: certain regions will not be either the stable or unstable manifold because the critical point is actually a saddle point on the surface; the surface solution here will have to be drawn out by extending other consistent stable or unstable manifolds to include this region. Some measure of goodness of fit of the solution would be needed based on how difficult it is to stitch the solution surfaces together.

4 CONCLUSIONS

Treating the shape from shading problem as an image dynamical system leads to new ideas for robust shape from shading algorithms. These algorithms are very parallel in character, and converge to integrable solution surfaces near the critical points without an external integrability condition.

Our results suggest integrability of a solution surface containing a critical point comes from the fact that it is an invariant smooth manifold containing the critical point. The algorithms based on the image dynamical system attempt to find these invariant manifolds directly. Integrability of the corresponding surface normals comes for free in the noise-free case, and can be used to monitor the progress of the algorithm in locating a reasonable solution when the image is noisy. The algorithm appears to be robust with respect to image noise and poor reflectance function choice.

Our theoretical results indicate that generically there will be at most four and at least two solution surfaces through a patch of an image containing a very good critical point due to a known reflectance function and a generic surface. These solutions correspond to the different invariant manifolds of the critical point seen as a critical point of the image dynamical system: the stable and unstable manifolds on which the image dynamical system has source and sink behavior, and potentially two other invariant manifolds on which the image dynamical system has saddle behavior. The critical points in the image for which this result holds are those due to local reflectance function maxima (the usual case) or minima.

An important question remaining is where the assumed reflectance function comes from. Several constraints on the reflectance function are available from the image dynamical system. There are the constraints on reflectance function critical points from the image brightness at good critical points. Brightnesses along the bounding contour (where the projection direction just grazes the surface) also provide information about the reflectance function because the surface orientations can be immediately determined from the image. (Image brightnesses near the bounding contour alone probably do not determine surface shape beyond the surface orientations along the bounding contour (Saxberg, 1989).) Also, as discussed earlier, in order for the image dynamical system to have a single solution surface, some subset of the invariant manifolds (including stable and unstable manifolds of convex and concave critical points) must merge rather than intersect in a one-dimensional curve. This is unstable behavior for generic dynamical systems, and so may provide a constraint on the choice of reflectance functions: although numerical errors will prevent exact matching of the invariant manifolds even with exact knowledge of the reflectance function, a wrong choice is likely to prevent the invariant manifolds from being even close together.

The modern methods of differential geometry provide a set of tools for reasoning about geometry and shape without always being tied to coordinate system expressions. They allow one to look at all the information available from the image and reflectance function in the image irradiance problem and see how properties of the dynamical system influence choices of possible solution surfaces; they can provide constraints on reflectance function choices as well. The theoretical view of the image irradiance problem as a dynamical system also suggests computational approaches to finding solution surfaces that are theoretically tractable.

There is an entire literature on dynamical systems developed over the last twenty years to study features of complicated dynamical systems. The mathematical tools were developed to help analyze geometric visualizations of complicated physical problems. It is perhaps time to use these tools to study the principles behind vision itself.

REFERENCES

Abraham, R., Marsden, J., *Foundations of Mechanics*, 2nd edition, Benjamin Cummings, 1985

Abraham, R., Marsden, J., Ratiu, T., *Manifolds, Tensor Analysis, and Applications*, Addison-Wesley, 1983

Brooks, M. J., "Shape from Shading Discretely," Ph.D. thesis, Department of Computer Science, Essex University, Colchester, England, 1982

Bruss, A., "The Image Irradiance Equation: Its Solution and Application," Ph.D. Thesis, Department of Electrical Engineering and Computer Science, MIT, 1980

Bülthoff, H. H., Mallot, H. A., "Interaction of Different Modules in Depth Perception," International Conference on Computer Vision, London, England, June 8-11, pp. 295-305, 1987

Deift, P., Sylvester, J., "Some Remarks on the Shape-from-Shading Problem in Computer Vision," Journal of Mathematical Analysis and Applications, 84:235-248, 1981

Frankot, R. T., Chellappa, R., "A Method for Enforcing Integrability in Shape from Shading Algorithms," International Conference on Computer Vision, London, England, June 8-11, pp. 118-127, 1987

Horn, B.K.P., "Obtaining Shape from Shading Information," Chapter 4 in *The Psychology of Computer Vision*, P.H. Winston (ed.), McGraw-Hill, pp. 115-155, 1975

Horn, B.K.P., Brooks, M. J., "The Variational Approach to Shape from Shading," Computer Vision, Graphics, and Image Processing, 33: 174-208, 1986

Ikeuchi, K., Horn, B.K.P., "Numerical Shape from Shading and Occluding Boundaries," Artificial Intelligence, 17: 141-184, 1981

Koenderink, J.J., Van Doorn, A. J., "Photometric Invariants Related to Solid Shape," Optica Acta, 27: 981-996, 1980

Mingolla, E., Todd, J. T., "Perception of Solid Shape from Shading," Biological Cybernetics, 53: 137-151, 1986

Palis, J., De Melo, W., *Geometric Theory of Dynamical Systems*, Springer-Verlag, 1982

Pentland, A. P., "The Visual Inference of Shape: Computation from Local Features," Ph.D. Thesis, Psychology Department, MIT, 1982

Pentland, A. P., "Local Shading Analysis," IEEE Transactions on Pattern Analysis and Machine Intelligence, PAMI-6: 170-187, 1984

Saxberg, B.V.H., "A Modern Differential Geometric Approach to Shape from Shading," Ph.D. Thesis, Department of Electrical Engineering and Computer Science, MIT, 1989

FINDING HIERARCHICAL CLUSTERS BY ENTROPY MINIMIZATION[1]

Richard S. Wallace and *Takeo Kanade*

School of Computer Science
Carnegie Mellon
Pittsburgh, PA 15213

Abstract

This paper reports a new hierarchical clustering algorithm called Numerical Iterative Hierarchical Clustering (*NIHC*). The algorithm uses a tree transformation operation called a ***grab*** to iteratively rearrange the nodes of a hierarchical cluster tree in order to reduce the value of an objective function defined over the tree. In particular this research investigates the application of the Gaussian entropy function to measure the quality of the cluster tree. The input to *NIHC* is an arbitrary cluster tree such as a *k-d* tree. *NIHC* repeatedly searches for grabs to transform each subtree into a lower entropy state. When *NIHC* can find no more energy-reducing grabs we claim that the resulting tree is partially optimal in a strong sense that does not generally hold for cluster trees produced by other algorithms such as agglomeration. We report experiments designed to quantitatively compare *NIHC* using Gaussian entropy with other hierarchical clustering algorithms. We also sketch a proof that to cluster n points an iteration of *NIHC* takes at most $O(n^3)$ steps for an unbalanced tree and at most $O(n^2)$ steps for a balanced tree. Unlike the agglomerative algorithm, *NIHC* does not store an $O(n^2)$ cluster distance matrix. *NIHC* consumes only $O(n)$ storage for the tree itself.

1. Introduction

Automatic clustering of real-valued vectors is critical to the solution of many computer vision problems including three-dimensional object localization by Hough transform [23] [32] [35], color image segmentation [19] [25] [29], "data fusion" of different sensor modalities [20][8], estimation of surface and texture [33], and perceptual grouping. Despite its wide applicability, some researchers consider cluster analysis to be an archaic research area. There are several reasons to think so. A few algorithms developed over 20 years ago prove effective in many simple situations. These algorithms, notably ISODATA, k-means, single linkage and complete linkage hierarchical clustering and Ward's method, work well when the problem size is small, when the clusters formed in the data are compact and well-separated, and when there is relatively little noise. The details of these algorithms have been studied extensively and their algorithmic complexity and optimality properties are well-known [7][9][15][17][18] [34][41]. These algorithms have also been subject to Monte Carlo experiments and a lot of "field testing" on real data. But practical experience applying these algorithms to vision problems has underscored their limitations. Our motivation to research cluster analysis was born out of the need to develop efficient, robust clustering procedures for the large-scale problems in computer vision.

We are investigating algorithms for hierarchical cluster analysis using the heuristic principle of entropy minimization. The application of entropy minimization to cluster analysis is not new. In a 1966 paper Williams and Lambert [42] defined an information gain measure for clustering objects with binary attributes. They cite the proceedings of a 1960 conference where a mention of entropy in clustering appeared even earlier. Wallace (not the author) and Boulton developed information measures for both level [38] and hierarchical [3] clustering. Jambu and Lebeaux [17] defined an entropy measure for clusters of objects with discrete-valued attributes. Elsewhere [39] we show that the entropy function H defined in section (2) below is almost equivalent to Jardine and Sibson's radius of information measure for normal populations[18][31]. Recently, Lee [22] used an entropy measure to cluster triphone models used in speech understanding. Cheeseman et al. [4] extended Wallace and Boulton's minimum description length formulation to solve the finite mixture problem for an unknown number of classes. Entropy minimization techniques in pattern recognition generally are widespread, considering "the methods widely used

[1] This research was supported in part by DARPA, monitored by the Air Force Avionics Lab under contract F33615-87-C-1499. Richard S. Wallace was supported in part by a Hughes Aircraft Company Doctoral Fellowship. The views and conclusions contained in this document are those of the authors and should not be interpreted as representing the official policies, either expressed or implied, of the funding agencies.

under [different] names, such as Karhunen-Loeve expansion, diagonalization of the covariance matrix, principal axis method, SELFIC and factor analysis *differ only in the emphasis placed on particular aspects of the same mathematical tool* and are based on the same theoretical structure. The heuristic principle of *entropy minimization* led to this mathematical tool" (page 199 of [41], italics added. See also [5]).

This paper reports experiments with and analysis of a new hierarchical clustering algorithm capable of reaching lower local entropy minima than the standard agglomerative algorithm. This algorithm, called *NIHC* (*N*umerical *I*terative *H*ierarchical *C*lustering), iteratively transforms a binary cluster tree to minimize an entropy measure.

This paper is organized in three main sections following the introduction. Section (2) formally specifies the domain of *NIHC* as well as the tree data structure it manipulates. Section (2) also defines the Gaussian entropy function and states some assumptions underlying the choice of Gaussian entropy for measuring the quality of hierarchical clusters of real-valued data. Section (3) describes the *NIHC* algorithm and its elementary transformation step, the *grab* operation, then describes the algorithm's worst-case complexity. Section (4) presents some experimental results with *NIHC*, comparing it with 7 other hierarchical clustering procedures on a standard data set and in Monte Carlo experiments.

2. Cluster trees and entropy minimization

The leaf node of a hierarchical cluster tree (or *clustree*) represents a singleton set containing one of the input points. In our case a data point $\mathbf{x} = (x_0, \ldots, x_{d-1})^T \in \Re^d$. The cluster trees described here are binary (specifically, B-trees).

In what follows let u be a node internal to a cluster tree. Let $l = u \rightarrow$*left* and $r = u \rightarrow$*right* be the left and right child nodes of t respectively. The set of points in the population represented by u is

$$\mathcal{S}(u) = \mathcal{S}(l) \cup \mathcal{S}(r) \quad \text{and} \quad \mathcal{S}(l) \cap \mathcal{S}(r) = \emptyset. \tag{1}$$

For the size $n(u) = |\mathcal{S}(u)|$ we have

$$n(u) = n(l) + n(r). \tag{2}$$

The general form for an energy function $e(u)$ on a clustree u is

$$e(u) = \begin{cases} 0 & \text{if } u = \text{NIL} \\ f(u) + e(l) + e(r) & \text{otherwise} \end{cases} \tag{3}$$

where $f(u)$ depends only on the set $\mathcal{S}(u)$. In many cases $f(u) = d(l, r)$ where $d(x, y)$ is a distance function between the nodes x and y. A hierarchical clustering procedure amounts to a search through the space of trees for one that minimizes the objective function $e(u)$.

A binary cluster tree having n leaf nodes has $n - 1$ internal nodes and therefore consumes $O(n)$ storage, provided the size of each node in memory is constant.

If we let

$$p = \frac{n(l)}{n(l) + n(r)} \tag{4}$$

then the mean vector of the set $\mathcal{S}(u)$ is

$$\mathbf{m}(u) = p\ \mathbf{m}(l) + (1-p)\ \mathbf{m}(r) \tag{5}$$

and its covariance matrix is

$$\mathrm{C}(u) = p\ \mathrm{C}(l) + (1-p)\ \mathrm{C}(r) + p(1-p)\ (\mathbf{m}(l) - \mathbf{m}(r))(\mathbf{m}(l) - \mathbf{m}(r))^T \tag{6}$$

So the size and first and second moments about the mean for the set $\mathcal{S}(u)$ represented by a node u in the cluster tree may be derived recursively from the size and first and second moment's of u's children

In general all the moments of $\mathcal{S}(u)$ may be derived recursively in the tree.

This research is specifically concerned with an energy function H defined as

$$H(t) = \begin{cases} 0 & \text{if } t = \text{NIL} \\ \log|\mathrm{C}(t)| + H(l) + H(r) & \text{otherwise,} \end{cases} \tag{7}$$

where $|\cdot|$ is the matrix determinant [1]. The term H is called *Gaussian entropy* because it is related to the *relative entropy* of a Gaussian [37] in the information-theoretic sense. That is, if $f(\mathbf{x})$ is a multivariate Gaussian pdf with covariance C, then the relative entropy

$$-\int_{\Re^d} f(\mathbf{x})\,\log f(\mathbf{x})\,dx \quad = \quad \tfrac{1}{2}\log|\mathrm{C}| + \tfrac{1}{2}\log 2\pi + \tfrac{1}{2}. \tag{8}$$

Shannon [30] solved the variational problem which shows that the expression on the right in equation (8) maximizes $-\int p(\mathbf{x})\log p(\mathbf{x})\,d\mathbf{x}$ for all continuous p having mean $\mathbf{m}$ and covariance C [2]. Each node u in a clustree represents a set $\mathcal{S}(u)$ which is a random sample from an unknown pdf $g(u,\mathbf{x})$. Finding the $g(u,\mathbf{x})$ is an ill-posed problem, but Shannon's result leads to a minimax principle: if we assume that $g(u,\mathbf{x})$ is continuous and that the sample mean $\mathbf{m}(u)$ and covariance $\mathrm{C}(u)$ are sufficient statistics for g's actual mean and covariance, then the quantity in equation (8) with $\mathrm{C} = \mathrm{C}(u)$ is an upper bound on the entropy of the true $g(u,\mathbf{x})$. Under the interpretation of entropy as a measure of "uncertainty", minimizing $H(t)$ amounts to minimizing the maximum uncertainty concerning the distributions $g(u,\mathbf{x})$. Note that while in principle knowing higher-order moments than $\mathbf{m}$ and C enables us to find a smaller upper bound on the entropy of $g(u,\mathbf{x})$, no bound derived from higher moments will be *larger* than the quantity in equation (8).

The representation of clusters by moments in the clustree data structure means that the clustering program solves line- and plane-fitting problems as a by-product of clustering. The mean value $\mathbf{m}(u)$ of a cluster $\mathcal{S}(u)$ is a point on the line (or plane) best fit to the cluster data $\mathcal{S}(u)$ the minimum squared-error sense. The eigenvector of $\mathrm{C}(u)$ with largest eigenvalue is its gradient. Also the eigenvalues of $\mathrm{C}(u)$ measure the spherical asymmetry of the cluster $\mathcal{S}(u)$.

3. The *NIHC* algorithm

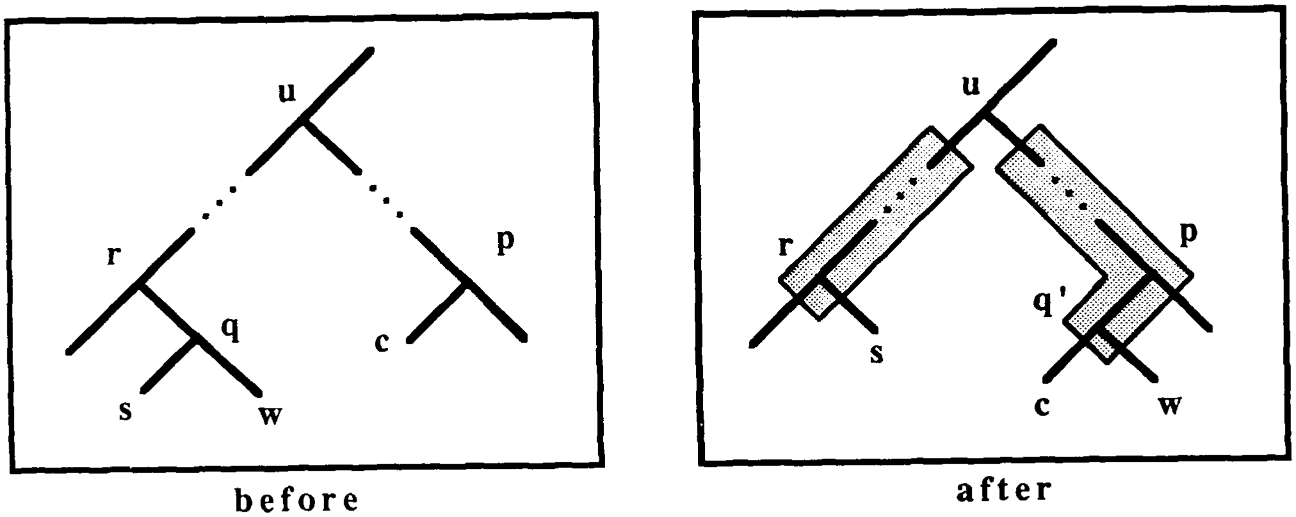
The name "Numerical Iterative Hierarchical Clustering" derives from the historical development of clustering algorithms. *NIHC* is a *hierarchical* clustering algorithm. Like other hierarchical algorithms, *NIHC* computes a cluster tree whose leaf nodes represent the input data points and whose internal nodes represent nested sets of those points. Unlike other hierarchical clustering algorithms, *NIHC* is neither divisive (top-down) nor agglomerative (bottom-up). Instead, it begins with an arbitrary binary cluster tree such as a *k-d* tree [2] and iteratively transforms it into a partially optimal cluster tree. Thus *NIHC* is *iterative* in a way that is analogous to partitioning methods like k-means [28], i.e. the result of each iteration is a clustering that is quantitatively improved over the previous iteration. Finally, we say that *NIHC* is *numerical* because of the type of data it clusters. *NIHC* clusters vectors in $\Re^d$, where d is the number of dimensions. Future research may uncover ways to write iterative hierarchical clustering algorithms for other types of data, such as ordinal and categorical data, but *NIHC* is defined only for real-valued data.

[1] One small problem arises with this definition. Unless a point set $\mathcal{S}(u)$ has at least $d+1$ points and is nondegenerate, its covariance matrix will be singular and $\log|\mathrm{C}(u)|$ won't exist. The solution is to treat leaf nodes not as points but as clusters with small variance. Specifically, a leaf node u representing a point $\mathbf{x}$ has $\mathbf{m}(u) = \mathbf{x}$ and $\mathrm{C}(u)$ set to some nonsingular value. This solution actually benefits us in another way, because it allows explicit representation of uncertainty in the data. For example, if the point $\mathbf{x}$ represented by u is measured from a sensor with known error $\mathrm{C}(\mathbf{x})$ then we set $\mathrm{C}(u) = \mathrm{C}(\mathbf{x})$.

[2] A more general result is that each member of the class of continuous maximum entropy pdfs has the exponential form and that this class is equivalent to the class of continuous pdfs admitting sufficient statistics (see [16][5][36])

3.1. The grab operation

The elementary transformation step in *NIHC* is a *grab*. Appel [1] defined a *grab* on a binary tree as follows. The operation *grab*(c, w) transforms a binary tree so that its nodes c and w become siblings. Let u be the first common ancestor of c and w. If $c = u$ or $w = u$ the grab is undefined. Otherwise, the grab makes c and w siblings by creating a new parent node q' with c and w as its children. q' takes the place of c in the original tree. Let q be the original parent of w and s be the original sibling of w. The grab makes s an only child. So s is "promoted" by deleting q and replacing it with s. The notation q and q' is intended to demonstrate that the grab operation conserves the number of nodes in the tree. Figure (3.1) illustrates the grab operation.



before

after

Figure 3.1. The grab operation

The effect of a grab in a clustree is to rearrange the nested subsets of points represented by nodes of the tree. Let u be the first common ancestor of c and w. The operation *grab*(c, w) rearranges sets as follows. If x is any node along the path from w's original grandparent r to u and x represents $\mathcal{S}(x)$ *before* the grab, then x represents $\mathcal{S}(x) - \mathcal{S}(w)$ *after* the grab. Similarly, if y is a node along the path from c's parent p to u and y represents $\mathcal{S}(y)$ before the grab, then after the grab y represents $\mathcal{S}(y) \cup \mathcal{S}(w)$. Figure (3.1). highlights the nodes representing different sets after a grab.

When the set represented by a node changes, so does the node's energy. But the grab only affects the nodes along paths back to the common ancestor. The energy of all other nodes in the tree remains unchanged. This observation, called the *grab property*, is key to writing *NIHC*. Because of the grab property *NIHC* can relatively efficiently calculate the effect of a particular *grab*(c, w), by calculating the energy change just along the path to the first common ancestor of c and w. *NIHC* works by calculating the net energy change of many grabs and choosing the best among them.

3.2. Definition and complexity of *NIHC*

Every subtree of a minimum energy clustree is a minimum energy clustree. That this fact holds is demonstrated by considering a counterexample. Suppose a minimum energy clustree t has a subtree s that is not minimum. We can replace s in t by a new tree u such that $\mathcal{S}(s) = \mathcal{S}(u)$ but $e(u) < e(s)$, i.e. s and u represent the same set of points but u has lower energy. But then t's energy is reduced, contradicting the assumption that t was minimum.

```
cluster ($t$)
 clustree *$t$
 begin
         clustree *$c$, *$w$
         if $t \neq$ NIL and $t \rightarrow$left $\neq$ NIL then
                 begin
                         cluster ($t \rightarrow$left)
                         cluster ($t \rightarrow$right)

                         find a pair of nodes $(c, w)$ such that
                                 grab $(c, w)$ reduces the energy of $t$
                                 more than any other grab

                         if $c \neq$ NIL and $w \neq$ NIL then
                                 grab $(c, w)$
                end
 end
```

Figure 3.2. Simplified version of *cluster*

This observation leads to a natural algorithm to search for a minimum energy clustree: one that tries to reduce the energy of each subtree. Numerical iterative hierarchical clustering (*NIHC*) works just that way, by searching each subtree for grabs that reduce its energy.

The input to *NIHC* is an arbitrary clustree t, such as a *k-d* tree or a clustree constructed by another hierarchical clustering algorithm. *NIHC* calls the procedure *cluster* iteratively on t. During each iteration, *cluster* looks for grabs to reduce the energy of t. *NIHC* calls *cluster* either (a) for a user-specified number of iterations or (b) until no more grabs can be found that reduce the energy of t. Under option (b), convergence (termination) is guaranteed because *cluster* either finds grabs that reduce t's energy or not. If it does not, no more iterations are possible. If *cluster* does find energy-reducing grabs, then the grabs reduce the energy of t. Since all the possible clustrees over the input data are ordered with respect to clustree energy, each iteration reduces the size of the set of trees in which t could be a member. Therefore, *NIHC* eventually terminates.

Figure (3.2) is a pseudocode description of the procedure *cluster*. By spelling out the details of *cluster* (and its subprocedures) we can analyze its worst-case complexity. But by examining figure (3.2) we can easily sketch the complexity argument. The procedure cluster visits every non-terminal subtree of the clustree t, looking for grabs that reduce the energy of the subtree. That *cluster* visits each subtree once is evident from its simple postorder recursive structure. The code hidden by the italic phrase "*find a pair of nodes* (c, w)..." basically works by comparing every node in one child tree of t with every node in the other. Thus, the cost accrued by *cluster* at each subtree s is $O(n(s)^2)$, where $n(s)$ is the size of the point set represented by s. The sum of the cost of *cluster* over all subtrees in t is largest when the tree t is a list (i.e. one child of each internal node is a leaf node), and in that case the sum is $O(n(t)^3)$. The sum is smallest when t is perfectly balanced (i.e. $n(l) = n(r)$ for each pair of siblings (l, r)), and that sum is $O(n(t)^2)$.

For comparison, the standard agglomerative algorithm (see [7] page 230) takes $O(n^3)$ steps for an n point clustering problem, but the time may be reduced to $O(n^2)$ or even to $O(n \log n)$ [17] for certain objective functions such as single-linkage. Typically the agglomerative procedure requires $O(n^2)$ storage for the cluster distance matrix but *NIHC* requires only $O(n)$ memory for the tree itself.

Also, *NIHC* outputs a tree satisfying a partial optimality condition not necessarily satisfied by the output of agglomeration. Given any pair of clustrees t and u, if $S(t) = S(u)$ then t may be constructed from u by some sequence of grab operations. So a tree t is *optimal* with respect to an objective function $e(t)$ provided there is no sequence of grabs that can reduce $e(t)$. We say t is *partially optimal* wrt $e(t)$ provided that no *single* grab can reduce $e(t)$. Generally the output of an agglomerative procedure is not partially optimal in this sense, but the output of *NIHC* always is.

4. Clustering experiments

It is difficult to evaluate clustering procedures objectively. If the goal of clustering is to find "natural clusters" then some subjectivity is required to determine whether one clustering is better than another. This fact has not stopped researchers from creating a variety of experiments to evaluate different clustering procedures, however [10][9]. The point of this section is to compare the results obtained with the algorithm *NIHC* using the entropy criterion with those obtained from "classical" hierarchical algorithms. The evaluation is based on clustering standard data (R. A. Fisher's iris data) and by repeating some previously published Monte Carlo experiments (the Kuiper-Fisher experiments) and a variation on the old experiments.

One type of "experiment" that appears throughout the clustering literature [27] [14] [24] [26] [40] [12] is a demonstration of the proposed clustering procedure applied to simple two-dimensional perceptual grouping problems, such as those illustrated in figure (4). These examples convey a qualitative sense of how well *NIHC* performs on "hard" clustering problems [3]. *NIHC* computed a clustree for each of the point patterns in the left columns and selected a subset of the clusters represented by the internal nodes. The selection is based on exploring the tree depth-first, taking the first cluster along each path from the root that has (a) small description length h (or equivalently small volume), and (b) a sufficiently large number of points n. Figure (4) displays the selected clusters in the right columns as plots of the 3 standard deviation elliptical contour of a Gaussian function having the same mean and covariance as the points in the cluster. The reported times are wall-clock times for a *C* implementation of *NIHC* on an unloaded Sun 4 computer.

In the experiments that follow we compare eight hierarchical clustering methods:

0.	agglomeration with entropy
1.	nearest neighbor (single linkage)
2.	furthest neighbor (complete linkage)
3.	median linkage
4.	group average (average linkage)
5.	centroid
6.	Ward's sum of squares
7.	NIHC with entropy

For a description of methods 1 through 6 see for example ([7] page 230ff). Each of the algorithms 0 through 6 is agglomerative and differs from the others only in the tree objective function. Algorithm 0 is the standard agglomerative algorithm using the Gaussian entropy function as a cluster distance function $d(l, r) = \log |C(t)|$ where t is a hypothetical parent node of l any r satisfying equations (2), (5) and (6).

[3] We collected 40 2-d point patterns qualitatively based on perceptual grouping problems published in the cluster analysis literature, 10 of which appear in figure (4). A copy of this data may be obtained by sending electronic mail to rsw@ius3.cs.cmu.edu

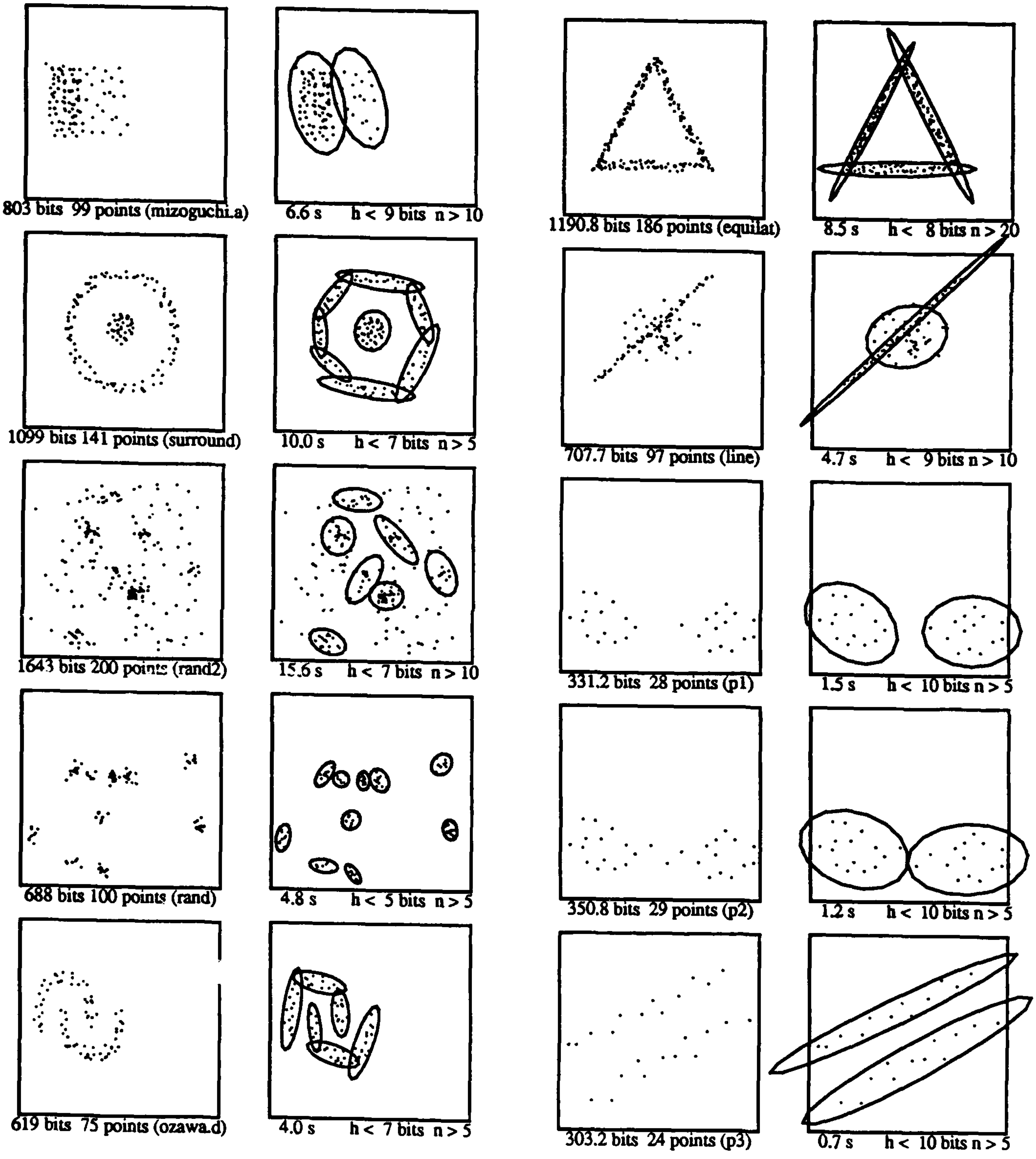

803 bits 99 points (mizoguchi.a)
6.6 s h < 9 bits n > 10
1190.8 bits 186 points (equilat)
8.5 s h < 8 bits n > 20
1099 bits 141 points (surround)
10.0 s h < 7 bits n > 5
707.7 bits 97 points (line)
4.7 s h < 9 bits n > 10
1643 bits 200 points (rand2)
15.6 s h < 7 bits n > 10
331.2 bits 28 points (p1)
1.5 s h < 10 bits n > 5
688 bits 100 points (rand)
4.8 s h < 5 bits n > 5
350.8 bits 29 points (p2)
1.2 s h < 10 bits n > 5
619 bits 75 points (ozawa.d)
4.0 s h < 7 bits n > 5
303.2 bits 24 points (p3)
0.7 s h < 10 bits n > 5

Figure 2: *NIHC* applied to perceptual grouping problems

4.1. The iris data

The data measured from the iris flower by E. Anderson and originally analyzed by R. A. Fisher in 1936 is well known to the pattern recognition and machine learning community [11][6][13][7][4]. The iris data consists of 3 classes of 50 4-d points each. Each class is derived from measurements of sepal and petal width and length from different types of iris flower: *setosa*, *versicolor* and *virgnica*. The setosa class is linearly separable from the other two. The virginica and versicolor classes are not linearly separable.

Owing to the small number of points, the input to *NIHC* is constructed by algorithm 0, agglomeration with entropy. In the case of *NIHC* the leaf cluster variance was set to $\sigma_{xx} = \sigma_{yy} = \exp(1/2)$ and $\sigma_{xy} = 0$. We selected three clusters from each clustree by splitting the root node into its two child clusters and then splitting the larger child into two more.

The following table shows the percentage of the $\binom{n}{2}$ point pairs that each procedure placed in the same cluster when the points came from the same class, or placed in different clusters when the points came from different classes. This measure, called "pair classification percentage" (PC%), is borrowed from the clustering experiments of Kuiper and Fisher [21].

	method							
	0	1	2	3	4	5	6	7
PC%	93.5	77.8	86.9	86.9	87.5	85.8	87.5	95.0

The PC% for the iris data is highest for *NIHC* minimizing entropy. In fact *NIHC* achieved 100% correct classification for the setosa class.

4.2. The Kuiper-Fisher experiments

Kuiper and Fisher compared the six hierarchical procedures 1 through 6 in a set of Monte Carlo experiments[21]. We duplicated their experiments, verified their results, and compared algorithms 0 and 7. Their original two cluster experiment was to cluster points from two unit-covariance bivariate Gaussian distributions with means $(0,0)$ and $(a,0)$. They randomly selected points from the distributions and ran agglomerative clustering on the point sets. Using the clusters corresponding to the left and right children of the root node of the clustree, they calculated the percentage of points classified correctly. Kuiper and Fisher repeated this experiment over 30 trials. They report percentages obtained for sets of 5, 10 and 15 points per class. The percentages are averaged with a varying from 0.5 to 3.5 in steps of 0.5. The input to *NIHC* is a *k-d* tree. As in the Iris experiment (see previous section), where appropriate the leaf cluster variance is set to $\sigma_{xx} = \sigma_{yy} = \exp(\frac{1}{2})$ and $\sigma_{xy} = 0$. The following table compares the results of the 8 algorithms with an ideal linear discriminant functior (in column L) calculated with the means and variances of the Gaussians known.

	method								
size	0	1	2	3	4	5	6	7	L
5	77.4	67.5	78.8	74.5	75.4	74.2	77.4	78.0	81.3
10	76.5	60.2	75.6	72.0	72.4	71.8	75.8	78.8	81.3
15	76.6	55.0	75.3	70.7	69.7	69.6	76.4	78.9	81.3
means	76.9	60.9	76.6	72.4	72.5	71.9	76.5	78.6	81.3

Stanard error for the reported values is ≤ 0.8. In this two-class problem *NIHC* with entropy performs at least marginally better on average than all other hierarchical methods, and agglomeration with entropy is in second place. Single linkage does not do well on this data due to "chaining", an effect arising from the close proximity of the two clusters causing the minimal spanning tree computed by single-linkage to "chain" from one cluster into the other.

Kuiper and Fisher measured the effect of different size clusters by considering two unequal sample sizes from unit-variance bivariate Gaussians. The following table shows the average percentage of points classified correctly by the 8 algorithms, following the same procedure as the first experiment but taking 15 sample points from one Gaussian and 5 from the other. In this case the averages are over 30 trials and d varying from 1 to 3.5 in steps of 0.5.

	method							
	0	1	2	3	4	5	6	7
%	68.7	66.2	68.4	72.1	72.0	70.5	68.3	70.0

The standard error is ≤ 0.1. This experiment illustrates a more difficult clustering problem for *NIHC*, but the new algorithm did not perform significantly worse than any other. Because of chaining single linkage also has problems with this data. For a discussion of the problems of clustering with unequal class size see ([7] page 217ff).

Our variation of the Kuiper-Fisher experiment considers the case of two clusters with equal means which may be distinguished by the orthogonality of their major axes. We used a sample of 30 points from each of two bivariate Gaussians having covariance matrices

$$\mathbf{C}_1 = \begin{pmatrix} \sigma_1^2 & 0 \\ 0 & \sigma_2^2 \end{pmatrix} \tag{9}$$

and

$$\mathbf{C}_2 = \begin{pmatrix} \sigma_2^2 & 0 \\ 0 & \sigma_1^2 \end{pmatrix} \tag{10}$$

respectively. In the trials $\sigma_1 = 1/(n+2)$ and $\sigma_2 = (n+2)$, with the results averaged over $n = 0, \ldots, 5$ in steps of 1.

The following table shows the results averaged over 30 trials.

	method							
	0	1	2	3	4	5	6	7
%	58.4	49.2	50.9	49.8	51.0	49.8	53.6	67.2

The standard error is ≤ 0.8. This final experiment highlights the ability of *NIHC* with entropy to cluster points from overlapping distributions. The clusters generated by other hierarchical algorithms are no better than random, but *NIHC* clustered an average of two thirds of the points correctly. This experiment also shows that the Gaussian entropy function in conjunction with the agglomerative algorithm (method 0) does not produce clusters as good as those generated by *NIHC* (method 7) using Gaussian entropy.

5. Discussion

Elsewhere [39] we discuss the details of the *NIHC* complexity argument and measure several methods for further speedups, such as a branch-and-bound procedure, heuristics to prune the grab search and parallel versions of *NIHC*. Also, we developed a program using *NIHC* that tracks time-varying clusters such as the data obtained from a robot range sensor in the presence of moving obstacles.

The largest clustering problem *NIHC* has so far solved consisted of 6400 2-d points, and it required 23 minutes on a single scalar processor of the Cray XMP. *NIHC* can solve smaller problems of 100 points from 1 to 5 dimensions in under 1 second of XMP time, using only one scalar processor. Profiling reveals that the program spends the largest share of its time (typically about 25%) computing log-determinants for the entropy calculation in the grab search. A vectorized or parallel implementation of the Gaussian entropy calculation will theoretically speed up the log-determinant computation from $O(d^3)$ steps on one processor to $O(d)$ on d^2 processors. We are investigating further parallel speedups in the *NIHC* routines.

The experiments and analysis presented in this paper warrant the following conclusions:

- *NIHC* using Gaussian entropy is competitive with other hierarchical clustering programs in speed and space requirements. In particular the algorithm's linear memory utilization means that it may be applied to large-scale cluster analysis problems such as 6-dimensional Hough transforms for object localization.
- In our experiments we quantitatively measured the ability to *NIHC* to find natural clusters and demonstrated in one case that *NIHC* using Gaussian entropy found natural clusters where the results of other hierarchical procedures were no better than random.
- *NIHC* never performed significantly worse than any agglomerative hierarchical algorithm in our tests.
- Future versions of *NIHC* running on hardware only 10 times faster than the Cray XMP will cluster small point patterns as fast as a sensor such as a scanning laser or a TV camera can measure them, fast enough to use clusters in a robot perception-control loop.

References

[1] Andrew W. Appel. *An Efficient Program for Many-Body Simulation (or, Cray Performance from a VAX)*. Computer Science Department Tech Report CMU-CS-83-118, Carnegie Mellon, 1983.

[2] J. L. Bentley. Multidimensional binary search trees used for associative searching. *Communications of the ACM*, 18(9), 1975.

[3] D. M. Boulton and C. S. Wallace. An information measure for single link classification. *The Computer Journal*, 18(3), 1975.

[4] Peter Cheeseman. AUTOCLASS II conceptual clustering system. In *Proceedings of the Machine Learning Conference*, pages 54–64, 1988.

[5] Ronald Christensen. *Entropy Minimax Sourcebook vol. 1*. Entropy Limited, Lincoln, MA, 1981.

[6] B. V. Dasarathy. Nosing around the neighborhood: a new system structure and classification rule for recognition in partially exposed environments. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, PAMI-2(1):67–71, 1980.

[7] Richard Duda and Hart. *Pattern Classification and Scene Analysis*. Wiley Interscience, 1973.

[8] Alberto Elfes and Larry Matthies. Sensor integration for robot navigation: combining sonar and stereo range data in a grid-based representation. In *IEEE Conference on Decision and Control*, 1987.

[9] Brian Everitt. *Cluster Analysis*. Heinemann Educational Books, 1974.

[10] Lloyd Fisher and John W. Van Ness. Admissible clustering procedures. *Biometrika*, 58, 1971.

[11] R. A. Fisher. The use of multiple measurements in taxonomic problems. *Annual Eugenics Part II*, 7:179–188, 1936.

[12] Frederick R. Fromm and Richard A. Northouse. CLASS: a nonparametric clustering algorithm. *Pattern Recognition*, 8:107–114, 1976.

[13] G. W. Gates. The reduced nearest neighbor rule. *IEEE Transactions on Information Theory*, 431-433, May 1972.

[14] K. Chidananda Gowda and G. Krishna. Agglomerative clustering using the concept of mutual nearest neighborhood. *Pattern Recognition*, 10:105–112, 1978.

[15] John A. Hartigan. *Clustering Algorithms*. John Wiley and Sons, 1975.

[16] Vasant S. Huzurbazar. *Sufficient Statistics: selected contributions*. Volume 19 of *Statistics: texbooks and monographs*, Dekker, 1976.

[17] M. Jambu and M.-O. Lebeaux. *Cluster Analysis and Data Analysis*. North-Holland, 1983.

[18] Nicholas Jardine and Robin Sibson. *Mathematical Taxonomy*. John Wiley and Sons Ltd, 1971.

[19] Gudrun J. Klinker. *The Measurement of Highlights in Color Images*. PhD thesis, Carnegie Mellon, 1988.

[20] Eric Krotkov. *Exploratory Visual Sensing for Determining Spatial Layout with an Agile Stereo Camera System*. PhD thesis, University of Pennsylvania, 1987.

[21] F. Kent Kuiper and Llyod Fisher. A monte carlo simulation of six clustering procedures. *Biometrics*, 31, 1975.

[22] Kai Fu Lee. *Large vocabulary speaker-independent continuous speech recognition: the SPHIX system*. PhD thesis, Carnegie Mellon, 1988.

[23] Seppo Linnainmaa, David Harwood, and Larry S. Davis. Triangle based pose determination of 3-d objects. In *8th International Conference on Pattern Recognition*, pages 116–8, 1986.

[24] Riichiro Mizoguchi and Masamichi Shimura. A nonparametric algorithm for detecting clusters using hierarchical structure. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, PAMI-2(4), 1980.

[25] Yu-ichi Ohta, Takeo Kanade, and Tohiyuki Sakai. Color information for region segmentation. *Computer Graphics and Image Processing*, 13, 1980.

[26] Kazumasa Ozawa. CLASSIC: a hierarchical clustering algorithm based on asymmetric similarities. *Pattern Recognition*, 16(2):201–211, 1983.

[27] F. James Rohlf. Adaptive hierarchical clustering schemes. *Syst. Zool.*, 19(1):58–82, 1970.

[28] Shokri Z. Selim and M. A. Ismail. K-means-type algorithms: a generalized convergence theorem and characterization of local optimality. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, PAMI-6(1), 1984.

[29] Steven A. Shafer and Takeo Kanade. Color vision. In *Encyclopedia of Artificial Intelligence*, Wiley-Interscience, 1984.

[30] C. E. Shannon. A mathematical theory of communication. *Bell System Technical Journal*, 27:379–423,623–656, 1948.

[31] Robin Sibson. Information radius. *Z. Wahrscheinlichkeitstheorie verw. Geb.*, 14:149–60, 1969.

[32] Teresa M. Silberberg, David A. Harwoord, and Larry S. Davis. Object recognition using oriented model points. *Computer Vision, Graphics, and Image Processing*, 35, 1986.

[33] J. F. Silverman. Bayesian clustering for unsupervised estimation of surface and texture models. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, PAMI-10(4), 1988.

[34] Peter H. A. Sneath and Robert R. Sokal. *Numerical Taxonomy.* W. H. Freeman and Company, 1973.

[35] D. W. Thompson and J. L. Mundy. Model-directed object recognition on the connection machine. In *DARPA IUS Workshop*, 1987.

[36] Myron Tribus. Thirty years of information theory. In Raphael D. Levine and Myron Tribus, editors, *The Maximum Entropy Formalism*, 1978.

[37] M. J. Usher. *Information Theory for Information Technologists.* MacMillan, 1984.

[38] C. S. Wallace and D. M. Boulton. An information measure for classification. *The Computer Journal*, 11:185, 1968.

[39] Richard S. Wallace. Finding natural clusters by entropy minimization. 1989. Ph.D. thesis in preparation.

[40] C. S. Warnekar and G. Krishna. A heuristic clustering algorithm using union of overlapping pattern-cells. *Pattern Recognition*, 11:85–93, 1979.

[41] Satosi Watanabe. *Pattern recognition: human and mechanincal.* North Holland, 1985.

[42] W. T. Williams and J. M. Lambert. Multivariate methods in plant ecology: v. similarity analyses and information analysis. *J. Ecology*, 54, 1966.

ROBUST COMPUTER VISION:
A LEAST MEDIAN OF SQUARES BASED APPROACH

Dong Yoon Kim[1]
J. John Kim
Peter Meer
Doron Mintz
Azriel Rosenfeld

Center for Automation Research
University of Maryland
College Park, MD 20742-3411

ABSTRACT

Regression analysis (fitting a model to noisy data) is a basic techniques in computer vision. Robust regression methods which remain reliable in the presence of various types of noise are therefore of considerable importance. We present a new paradigm based on the least median of squares (LMS) method proposed by Rousseeuw (1984). The method yield the correct result even when 49.9% of the data is severely corrupted. Its efficiency in the presence of Gaussian noise can be improved by complementing it with a weighted least squares based procedure. The high time complexity of the LMS algorithm can be reduced by a Monte Carlo type speed up technique. The algorithm was successfully applied to mode-based cluster detection, line fitting to noisy data, and designing a local operator performing robust plane fitting in images.

1. INTRODUCTION

Regression analysis (fitting a model to noisy data) is an important statistical tool frequently employed in computer vision for a large variety of tasks. Tradition and ease of computation have made the least squares method the most popular form of regression analysis. The least squares method achieves the optimum results when the underlying error distribution is Gaussian. However, the method becomes unreliable if the noise has non-Gaussian components and/or if outliers (samples with values far from the local trend) are present in the data. The outliers may be the result of clutter, large measurement errors, or impulse noise corrupting the data. At a transition between two homogeneous regions of the image, samples belonging to one region may become outliers for fits to the other region.

Three concepts are usually employed to evaluate a regression method: relative efficiency, breakdown point, and time complexity. The *relative efficiency* of a regression method is defined as the ratio between the lowest achievable variance for the estimated parameters (the Cramer-Rao bound) and the actual variance provided by the given method. The efficiency also depends on the underlying noise distribution. For example, in the presence of Gaussian noise the mean estimator has an asymptotic (large sample) efficiency of 1 (achieving the lower bound) while the median estimator's efficiency is only $\frac{2}{\pi} = 0.637$ (Mosteller and Tukey, 1977).

The *breakdown point* of a regression method is the smallest amount of outlier contamination which may force the value of the estimate outside an arbitrary range. For example, the breakdown point of the mean is 0 since a single large outlier can corrupt the result. The median remains unchanged if less than half of the data are contaminated, yielding asymptotically the maximum breakdown point, 0.5.

The *time complexity* of the least squares method is $O(np^2)$ where n is the number of data points and p is the number of parameters to be estimated. Feasibility of the computation requires a time complexity of at most $O(n^2)$.

A new, improved regression method should provide:

[1] Science Exchange Program Fellow from the Agency for Defense Development, Taejeon, Korea

- reliability in the presence of various types of noise, i.e., good asymptotic and small sample efficiency;
- protection against a high percentage of outliers, i.e., a high breakdown point;
- a time complexity not much greater than that of the least squares method.

Many statistical techniques have been proposed which satisfy some of the above conditions. These techniques are known as ***robust regression*** methods. In Section 2 a review of robust regression methods is given. In Section 3 the least median of squares (LMS) method is discussed in detail. This method is then applied to several computer vision problems: in Section 4 to mode-based cluster detection; in Section 5 to line fitting; and in Section 6 to noise cleaning in images through robust local plane fitting.

2. ROBUST REGRESSION METHODS

The early attempts to introduce robust regression methods involved straight line fitting. In one class of methods the data is first partitioned into two or three nearly equal sized parts ($i \leq L\,;\, L < i \leq R\,;\, R < i$) where i is the index of the data and $L = R$ in the former case. The slope β_1 and the intercept β_0 of the line are found by solving the system of nonlinear equations

$$\begin{aligned} \underset{i \leq L}{\text{med}} \,(\, y_i - \beta_0 - \beta_1 x_i \,) &= \underset{i > R}{\text{med}} \,(\, y_i - \beta_0 - \beta_1 x_i \,) \\ \underset{\text{for all } i}{\text{med}} \,(\, y_i - \beta_0 - \beta_1 x_i \,) &= 0 \end{aligned} \tag{1}$$

where med represents the median operator applied to the set defined below it. The breakdown point of the method is $0.5/k$ where k is the number of partitions (2 or 3) since the median is used for each part separately. Brown and Mood (1951) investigated the method for $k = 2$, and Tukey introduced the resistant line procedure for $k = 3$ (see Johnstone and Velleman, 1985).

Another class of methods uses the slopes between each pair of data points without splitting up the data set. Theil (1950) estimated the slope as the median of all $n(n-1)/2$ slopes which are defined by n data points. The breakdown point of these methods is 0.293 since at least half the slopes should be correct in order to obtain the correct estimate. That is, if ϵ is the fraction of outliers in the data we must have $(1-\epsilon)^2 \geq 0.5$. The intercept can be estimated from the input data by employing the traditional regression formula.

The theory of multidimensional robust estimators was developed in the 70's. The basic robust estimators are classified as M-estimators, R-estimators and L-estimators (Huber, 1981).

The M-estimators are the most popular robust regression methods. These estimators minimize the sum of a symmetric, positive-definite function ρ of the residuals r_i. (A residual is defined as the difference between the data point and the fitted value.) For the least squares method $\rho(r_i) = r_i^2$. The M-estimates of the parameters are obtained by iteratively solving the minimization problem

$$\min \sum_i \rho(\, r_i \,) \tag{2}$$

Notice that the sought parameters are represented through the residuals. Several ρ functions have been proposed. Huber (1981) employed the squared error for small residuals and the absolute error for large residuals. Andrews (1974) used a squared sine function for small and a constant for large residuals. Beaton and Tukey's (1974) biweight is another example of these ρ functions. Holland and Welsch (1977) developed algorithms for solving the numerical problems associated with M-estimators.

R-estimators are based on ordering the set of residuals. Jaeckel (1972) proposed obtaining the parameter estimates by solving the minimization problem

$$\min \sum_i a_n(R_i)\, r_i \tag{3}$$

where r_i is the residual; R_i is the location of the residual in the ordered list, i.e., its rank; and a_n is a score function. The score function must be monotonic and $\sum_i a_n(R_i) = 0$. The most frequently used score function is that of Wilcoxon: $a_n(R_i) = R_i - (n+1)/2$. Since $|\,a_n(R_i)| \leq (n-1)/2$ the largest residuals caused by outliers cannot have a too large weight. Scale invariance (independence from the variance of the noise) is an important advantage of R-estimators over M-estimators. Cheng and Hettmansperger (1983) presented an iteratively reweighted least squares algorithm for solving the minimization problem associated with R-estimators.

The L-estimators employ linear combinations of order statistics. The median and α-trimmed mean based methods belong to this class. It is important to notice, however, that the mean ($\alpha = 0$) is a least squares estimate, while the median can be regarded also as the M-estimate obtained for $\rho = |r_i|$. Various simulation studies have shown that L-estimators give less satisfactory results than the other two classes (Heiler, 1981).

In spite of their robustness for various distributions the M-, R- and L-estimators have breakdown points that are less than $1/(p+1)$, where p is the number of parameters in the regression (Li, 1985). For example, in planar surface fitting we have $p = 3$, and the breakdown point is less than 0.25, making it sensitive to outliers.

Recently several robust estimators having breakdown points close to 0.5 were proposed. Siegel (1982) introduced the repeated median (RM) method of solving multidimensional regression problems. Suppose p parameters are to be estimated from n data samples. A parameter is estimated in the following way: First, for each possible p-tuple of samples the value of the parameter is computed yielding a list of $C(n,p)$ (the binomial coefficient) terms. Then the medians for each of the p indices characterizing a p-tuple are obtained recursively. When the list has collapsed into one term, the result is the RM estimate of the parameter. Once a parameter has been estimated, the amount of computation can be reduced for the remaining $p-1$ parameters.

For example, let $p = 3$ and suppose that we start by estimating the parameter β_2 of the planar fit

$$z = \beta_0 + \beta_1 x + \beta_2 y \tag{4}$$

First the values

$$\beta_2(i,j,k) = \frac{(z_i - z_j)(x_j - x_k) - (z_j - z_k)(x_i - x_j)}{(y_i - y_j)(x_j - x_k) - (y_j - y_k)(x_i - x_j)} \tag{5}$$

are computed for all the triplets defined by $i \neq j \neq k$. The estimate is then

$$\hat{\beta}_2 = \underset{i}{\text{med}}\ \underset{j(\neq i)}{\text{med}}\ \underset{k(\neq i,j)}{\text{med}}\ \beta_2(i,j,k) \tag{6}$$

The parameter β_1 is estimated next, by applying the same algorithm for $p = 2$ to the data $z_i - \hat{\beta}_2 y_i$. Similarly the value of β_0 is obtained by taking the median of the samples $z_i - \hat{\beta}_2 y_i - \hat{\beta}_1 x_i$. The breakdown point of the repeated median method is 0.5 since all the partial median computations are performed over the entire data set. Computation of the median is $O(n)$ and thus the time complexity of the RM method is high, of $O(n^p)$ order. The Gaussian efficiency of the method was found experimentally as being only around 0.6 (Siegel, 1982). The τ-estimate introduced by Yohai and Zamar (1988) achieves high efficiency and high breakdown point simultaneously but its time complexity is very high.

The least median of squares robust regression method proposed by Rousseeuw(1984) also achieves 0.5 breakdown point. The relative efficiency of the method can be improved by combining it with least squares based techniques. The time complexity can be reduced by a Monte Carlo type speed-up technique. In the next section we describe the LMS method in detail.

3. THE LEAST MEDIAN OF SQUARES METHOD

Rousseeuw (1984) proposed the least median of squares (LMS) method in which the parameters are estimated by solving the nonlinear minimization problem

$$\min \underset{i}{\text{med}}\ r_i^2 \tag{7}$$

That is, the estimates must yield the smallest value for the median of residuals computed for the entire data set. The precise meaning of the minimization is clarified below. As in the case of the repeated median method, the LMS minimization problem (7) is also solved by a search in the space of possible estimates generated from the data.

Let a distinct p-tuple of data points be denoted by the indices $i_1, \ldots, i_p$. For every p-tuple the values of $p-1$ parameters (all except the intercept, β_0) are computed. The intercept is chosen as variable because it the *easiest to manipulate in computations*. There are $C(n,p)$ p-tuples and the minimization is performed in two steps. First, for a given p-tuple the intercept value $\beta_0(i_1, \ldots, i_p)$ that solves the minimization problem

$$\min \underset{i}{\text{med}}\ r_i^2 \qquad \text{given} \quad \beta_j(i_1, \ldots, i_p),\ j = 1, \ldots, (p-1). \tag{8}$$

is obtained. The procedure is repeated for every p-tuple and the one yielding the smallest value for (8) supplies the LMS estimates of *all* the parameters.

Steele and Steiger (1986) proposed to solve (8) by a mode estimation technique. (The mode of a histogram, i.e., probability distribution, is the location of its largest value.) Let $s_1 \leq s_2 \leq \cdots \leq s_n$, be a sorted list of values obtained from the data. To locate the mode of the underlying continuous distribution the *minimum* of the differences

$$D_i = s_{i+\lfloor n/2\rfloor} - s_i \qquad i = 1,2,\ldots,\lceil n/2\rceil \tag{9}$$

is sought. The functions $\lfloor\cdot\rfloor$ and $\lceil\cdot\rceil$ are the floor and ceiling functions respectively. The minimum difference will appear where most of the values are similar and since the list was sorted this coincides with the peak of the distribution. The resolution of the method is controlled by the distance between the two samples (9) and local maxima are avoided by choosing the largest possible distance $\lfloor n/2\rfloor$.

Assume that the mode was found to correspond to $i = k$. Its value is then taken as

$$M_k = (s_{k+\lfloor n/2\rfloor} + s_k)/2 \tag{10}$$

The minimum difference can be written as function of M_k:

$$D_k = M_k - s_k = s_{k+\lfloor n/2\rfloor} - M_k \tag{11}$$

and the following ordering relations are also valid:

$$|M_k - s_i| \begin{cases} \le D_k & \text{if } k < i < k+\lfloor n/2\rfloor \\ \ge D_k & \text{if } 1 \le i < k \text{ or } k+\lfloor n/2\rfloor < i \le n \end{cases} \tag{12}$$

because the s_i are ordered. From (11) and (12) we obtain

$$D_k = \operatorname*{med}_i |M_k - s_i| \tag{13}$$

since $\lfloor n/2\rfloor$ difference values are always less than or equal to D_k while the same number of differences are larger than or equal to it.

When the above procedure is performed for the n intercept values computed for a given p-tuple of data points, the resulting minimum difference is the solution of the minimization problem (8). For n data points $C(n,p)$ minimum differences $D_k(i_1,\ldots,i_p)$ are obtained and the smallest one yields the LMS estimate for the p parameters, i.e. the solution of (7). A detailed example of the LMS algorithm is given in Section 5.

The breakdown point of the least median squares method is 0.5 because all the median computations are over the whole data set. The time complexity of the method, however, is very high. There are $O(n^p)$ p-tuples and for each of them the sorting takes $O(n \log n)$ time. Thus the amount of computation required for the basic LMS algorithm is $O(n^{p+1}\log n)$, prohibitively large. Notice that this complexity is valid only if $p \ge 2$, since for $p = 1$ only sorting is required.

The time complexity is reduced to practical values when a Monte Carlo type speed-up technique is employed in which a $Q \ll 1$ probability of error is tolerated. Let ϵ be the fraction of data contaminated by outliers. Then the probability that all m different p-tuples chosen at random will contain at least one or more outliers is

$$P = [1 - (1-\epsilon)^p]^m \tag{14}$$

Note that $1 - P$ is the probability that at least one p-tuple from the chosen m has only uncorrupted samples and thus the correct parameter values can be recovered. The smallest acceptable value for m is the solution of the equation

$$P = Q \tag{15}$$

rounded upward to the closest integer, and is independent of n, the size of the data. The amount of computation becomes $O(m\ n \log n)$. This time complexity reduction is very significant. In Table 1 the values of m are given for three values of Q, p between 2 and 8, and ϵ between 0.05 and 0.499. For example, if $p = 3$, $Q = 0.01$ and $\epsilon = 0.3$ then $m = 11$ for any n. Thus, when at most 30 percent of the data is contaminated by outliers, by choosing 11 triplets for the computation of the LMS robust planar fit, the probability of having the whole set of triplets corrupted is 0.01.

Table 1: Number of p-tuples required to achieve the probability of error Q as a function of ϵ, the fraction of outliers.

$Q = 0.01$

$p \setminus \epsilon$	0.05	0.10	0.15	0.20	0.25	0.30	0.35	0.40	0.45	0.499
2	2	3	4	5	6	7	9	11	13	16
3	3	4	5	7	9	11	15	19	26	35
4	3	5	7	9	13	17	24	34	48	71
5	4	6	8	12	17	26	38	57	90	144
6	4	7	10	16	24	37	59	97	165	289
7	4	8	12	20	33	54	92	163	301	579
8	5	9	15	26	44	78	143	272	548	1158

$Q = 0.005$

$p \setminus \epsilon$	0.05	0.10	0.15	0.20	0.25	0.30	0.35	0.40	0.45	0.499
2	3	4	5	6	7	8	10	12	15	19
3	3	5	6	8	10	13	17	22	30	40
4	4	5	8	11	14	20	27	39	56	82
5	4	6	10	14	20	29	43	66	103	166
6	4	7	12	18	28	43	68	111	189	333
7	5	9	14	23	37	62	106	187	346	667
8	5	10	17	29	51	90	164	313	631	1333

$Q = 0.001$

$p \setminus \epsilon$	0.05	0.10	0.15	0.20	0.25	0.30	0.35	0.40	0.45	0.499
2	3	5	6	7	9	11	13	16	20	24
3	4	6	8	10	13	17	22	29	38	52
4	5	7	10	14	19	26	36	50	72	107
5	5	8	12	18	26	38	57	86	134	216
6	6	10	15	23	36	56	89	145	247	434
7	6	11	18	30	49	81	138	244	451	869
8	7	13	22	38	66	117	214	408	822	1737

When Gaussian noise is present in addition to outliers the relative efficiency of the LMS method is low. Rousseeuw (1984) has shown that the LMS method converges for large sample sizes as $n^{-1/3}$, much slower than the usual $n^{-1/2}$ for maximum likelihood estimators. To compensate for this deficiency he proposed combining the LMS method with a weighted least squares procedure which has high Gaussian efficiency. Either one-step weighted least squares or an M-estimator with Hampel's redescending function can be employed. For more detail see the book of Rousseeuw and Leroy (1987).

The breakdown point of the combined method is still 0.5 since the standard deviation of the noise, σ, is estimated from the LMS part and thus the weights in the least squares procedure can be correctly determined. The standard deviation estimate

$$\hat{\sigma} = 1.4826 \left(1 + \frac{5}{n-p} \right) \underset{i}{\text{med}} \, | r_i | \tag{16}$$

can be immediately obtained since the median of the residual is the value returned by the LMS procedure for the parameter estimates. Note that the usual robust standard deviation estimate does not contain the term $5/(n-p)$. This term is recommended by Rousseeuw and Leroy (1987) as a finite sample correction factor.

In the following sections we describe different applications of the LMS technique to computer vision problems. The number of estimated parameters p increases from one in mode-based clustering, to three in the computation of local planar fits.

4. MODE-BASED CLUSTER DETECTION

Given n data points in a plane, finding the centers of their clusters is a classical problem in pattern recognition. The number of clusters K is usually known a priori. Most clustering algorithms have four steps (see Jain and Dubes (1988) for a monograph on the subject):

Step 1. The n points are initially partitioned arbitrarily into K groups.

Step 2. The center of each group is estimated.

Step 3. The points are re-partitioned based on their distances to the current cluster centers.

Step 4. Steps 2 and 3 are repeated until the changes no longer exceed a convergence threshold.

The weight center of the points belonging to a cluster is often taken as its center, i.e., the mean of the x and y coordinates of the points. This method is known as the K-means clustering technique. The K-means approach, however, introduces artifacts whenever the data points are corrupted by non-uniformly distributed noise. The time complexity of the K-means technique is $O(n)$.

Another basic method of cluster detection involves searching for regions of high density—that is, mode seeking. In the mode seeking approach uniformly distributed noise points do not offset the mode of the original data unless the noise destroys the data entirely. Two classes of mode seeking methods are described in the literature. In the first class the points are grouped into bins and local maxima of the resulting multi-dimensional histogram are sought. The method is very sensitive to bin size; too small a size yields false alarms, while too large a size may smooth out significant maxima. The method requires large storage space, but its time complexity is still $O(n)$. In the other class of mode seeking methods the distances between all the possible point pairs are taken into consideration for clustering. These methods have the time complexity $O(n^2)$.

In Section 3 we have shown that mode estimation is part of the least median of squares algorithm and therefore application of the algorithm to mode-based cluster detection is immediate. As the examples will show, cluster detection using the LMS method accurately locates the cluster centers even in non-uniform background noise.

Assume that after the $(l-1)$st iteration the n data points were partitioned into K clusters having centers $[\, x_{c_i}(l-1),\, y_{c_i}(l-1) \,]$, where $i = 1,2, \ldots, K$. The ith cluster contains $n_i(l)$ data points.

The lth iteration of the clustering algorithm starts by independently computing the updated x and y coordinates for the new cluster centers. The same LMS based procedure is employed. Let $x_1, \ldots, x_{n_i(l)}$ be the list of abscissas for the points currently belonging to the ith cluster. After the application of the LMS algorithm ($p = 1$ in this case), the mode of the underlying distribution (9) is taken as a first approximation to the abscissa of the ith cluster center, $X_{c_i(l)}$. The robust standard deviation $\hat{\sigma}_{i,x}(l)$ (16) of this one-dimensional distribution is also estimated.

One-step weighted least square fitting is performed next for each two-dimensional cluster to increase the relative efficiency of the clustering algorithm to Gaussian noise. The distance between the jth data point (x_j, y_j) belonging to the ith cluster and the current center of the cluster is defined as:

$$d_{i,j}^2(l) = \left(\frac{x_j - X_{c_i(l)}}{\hat{\sigma}_{i,x}(l)} \right)^2 + \left(\frac{y_j - Y_{c_i(l)}}{\hat{\sigma}_{i,y}(l)} \right)^2 \tag{17}$$

At the initial partition, the directional standard deviations are set to one. Hence, (17) gives the Euclidean distance initially, and gives normalized distances at subsequent partitions. Notice that we assumed that each cluster has an elliptical shape with the major axes parallel to one of the sides of the image. For rotated clusters a cross-term including the correlation coefficient estimated from the data should be added to (17). Based on its distance the data point can be allocated with the weight $w_j(l)$:

$$w_j(l) = \begin{cases} 1 & d_{i,j} \leq 2.96 \\ 0 & d_{i,j} > 2.96 \end{cases} \tag{18}$$

where the threshold 2.96 corresponds to 98.76 percent of the two-dimensional normal distribution being taken as inliers. The coordinates $[\,x_{c_i}(l),\, y_{c_i}(l)\,]$ of the ith cluster center after the lth iteration are then the solutions of the least squares problems

$$\min_{x_{c_i(l)}} \sum_{j=1}^{n_i(l)} w_j(l)\,[\,x_j - x_{c_i(l)}\,]^2 \qquad \min_{y_{c_i(l)}} \sum_{j=1}^{n_i(l)} w_j(l)\,[\,y_j - y_{c_i(l)}\,]^2 \tag{19}$$

To further increase the Gaussian efficiency of the clustering algorithm the directional standard deviations are reestimated from the two-dimensional data taking into account only the inliers of the cluster:

$$\hat{\sigma}_{i,x}(l) = \frac{\sum_{j=1}^{n_i(l)} w_j(l)\,[\,x_j - \overline{x}_i\,]^2}{\sum_{j=1}^{n_i(l)} w_j(l) - 1} \qquad \hat{\sigma}_{i,y}(l) = \frac{\sum_{j=1}^{n_i(l)} w_j(l)\,[\,y_j - \overline{y}_i\,]^2}{\sum_{j=1}^{n_i(l)} w_j(l) - 1} \tag{20}$$

where $\overline{x}_i$ and $\overline{y}_i$ are the mean coordinates of the ith cluster's inlier points.

Once the updated coordinates of all the K cluster centers are found the new, $(l+1)$st partitioning of the data points can be performed. The partitioning is done by finding the closest cluster center for each point. The distance (17) is computed from a given point to every cluster center, employing the new directional standard deviations of the given cluster. The point is allocated to the cluster yielding the smallest distance. The $(l+1)$st iteration can now start. The iterations are repeated until no change occurs between two consecutive partitions.

Since only one parameter (the abscissa or the ordinate) is estimated by the LMS algorithm the time complexity is given by the sorting of the data points and is $O(n \log n)$ per iteration. Putting the restriction on the data that every point lie on an integer grid of a fixed size, which is the most common case in computer vision, the sorting can be accomplished in linear time. The complexity of the LMS algorithm is then only $O(n)$, the same as the K-means algorithm mentioned at the beginning of the section.

To compare the LMS based clustering method with other techniques three typical test data sets were used: circularly symmetric clusters (CSC), CSC's in uniform background noise, and CSC's in non-uniform background noise. Three cluster detection methods were applied to each data set. Besides the K-means and K-LMS methods the K-weighted-means technique proposed by Jolion and Rosenfeld (1988) was also investigated. In this method each data point is given a weight according to the density of points in its vicinity and thus the time complexity is $O(n^2)$.

Figure 1a shows the first data set, two CSC's having 100 and 150 points, distributed as Gaussians with means (10, 10) and (35, 35) and standard deviations 5. The experimental results are given in Table 2.

Table 2: Detected Cluster Centers. Noiseless case.

Method	Center 1	Center 2
K-means	(10.01, 10.06)	(35.06, 34.74)
K-weighted-means	(10.77, 10.93)	(35.14, 34.92)
K-LMS	(9.86, 9.98)	(35.01, 34.81)

As expected, Gaussian clusters are best estimated by the maximum likelihood K-means method but the other two algorithms also give correct answers.

In Figure 1b 75 uniformly distributed points were added as background noise to t set.

Table 3: Detected Cluster Centers. Uniform Background Noise.

Method	Center 1	Center 2
K-means	(11.11, 11.49)	(34.86, 33.46)
K-weighted-means	(9.85, 9.90)	(35.14, 34.70)
K-LMS	(10.55, 10.33)	(35.17, 34.76)

As shown in Table 3, only the result of K-means is significantly affected by the noise.

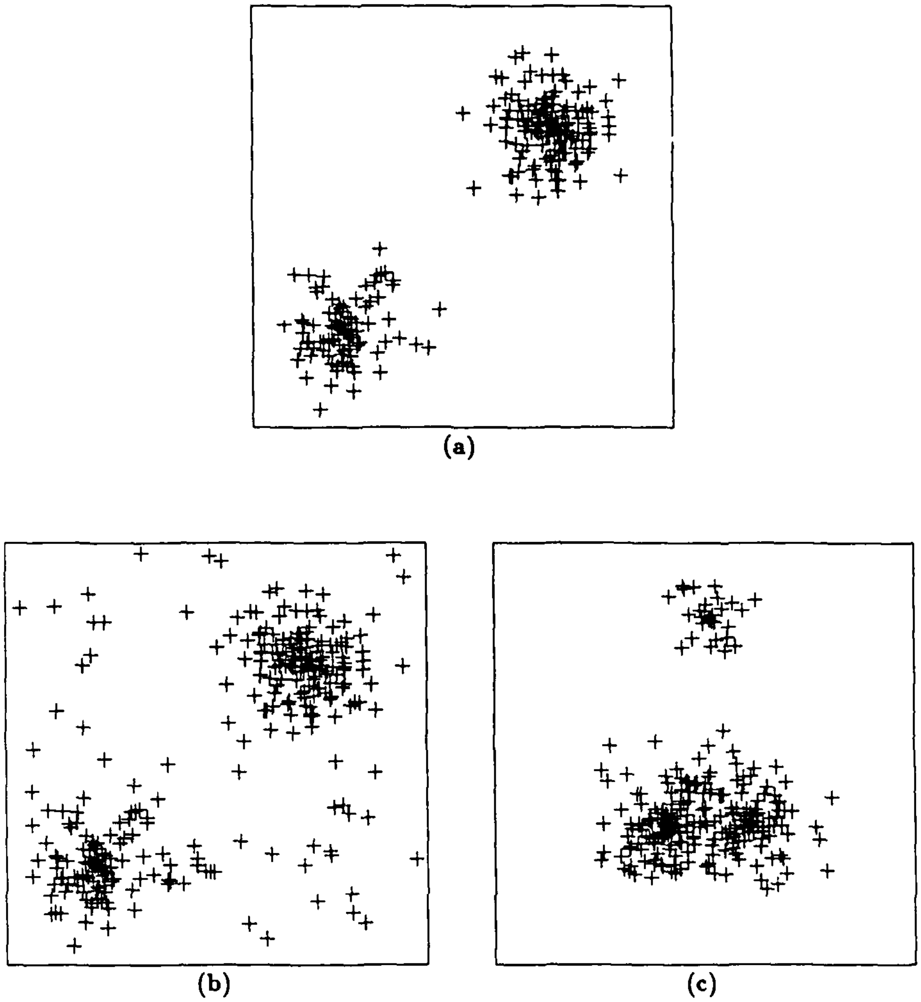


Figure 1. Test data for the clustering experiments. a) Two circularly symmetric clusters. b) The clusters from a) embedded in uniform background noise. c) Two data clusters with a biased noise cluster (top).

Figure 1c shows two CSC's having 100 and 150 points, distributed as Gaussians with means (20, 15) and (30, 15) and standard deviations 5, together with 30 biased noise points also normally distributed around (25, 40) with standard deviation 3.

Table 4: Detected Cluster Centers. Non-uniform Background Noise.

Method	**Center 1**	**Center 2**
K-means	(23.95, 15.26)	(25.22, 40.18)
K-weighted-means	(20.00, 14.91)	(28.84, 18.21)
K-LMS	(20.59, 15.06)	(30.25, 15.28)

The results given in Table 4 show that only the K-LMS method gives the correct result. The noise can also be regarded as a cluster not taken into account, i.e., the a priori information about the number of clusters was wrong. In this case the output of the K-LMS method can be employed for detecting the overlooked cluster. Subtracting the result from the original data and performing a clustering on the difference the third cluster is detected.

In our experiments with the K-LMS method the data points were restricted to a lattice with unit step size, i.e., the coordinates of a point were rounded to the nearest integers. While this coarse quantization was not present in the other two methods the LMS based algorithm's performance was always equal or superior to them. The number of storage bins required by the K-LMS algorithm increases only linearly with the dimension of the feature vectors (two in our examples) since the modes are determined separately along each coordinate axis. This is another advantage of the K-LMS algorithm relative to the histogram based methods, in which the storage increases exponentially with the dimension of the feature vector.

5. LINE FITTING TO NOISY DATA

Detection of straight lines in noisy data containing fragmented segments is an important task in computer vision. The Hough transform, one of the most often employed methods, can already be classified as a robust technique since it has the ability to detect the longest line segment even if it comprises less than 50% of the data points. For purposes of comparison with the LMS based line fitting method to be described below we consider the *pairwise* variant of the Hough transform in which for each pair of data points (x_i, y_i) and (x_j, y_j) the values of the parameter pair (ρ, θ) are calculated from the equations

$$\begin{aligned} \rho &= x_i \cos\theta + y_i \sin\theta \\ \rho &= x_j \cos\theta + y_j \sin\theta \end{aligned} \tag{21}$$

A histogram in (ρ, θ) space is built for all combinations of pairs of points. The line segments are detected by finding the peaks of the histogram, i.e., its modes.

The disadvantage of any type of Hough transform method is due to the histogram usage. A histogram using a bin size that is too small may have a wrong mode, while a histogram using a bin size that is too large may yield estimates that are too coarse. The discretization of image space also makes the distribution of parameters in Hough space non-homogeneous and non-equiprobable. The time complexity of the pairwise Hough transform is $O(n^2)$.

In the previous sections we have shown how the least median of squares algorithm finds the mode of a distribution, and how this algorithm can be employed for clustering problems. The histogram bin size does not create artifacts for the K-LMS clustering method, since the resolution of its mode seeking procedure is controlled only by the distance employed when computing the differences (11) between the sorted samples. For n data points, however, the K-LMS algorithm requires an additional $O(n^2 \log n)$ processing time to detect the clusters in the Hough space.

Before proceeding to give the details of the LMS based line fitting method we will prove that a speed-up technique similar to the one described in Section 3 cannot be successfully applied to the pairwise Hough transform. For example, the number of sample pairs which can guarantee a 0.01 error probability for a speeded-up LMS line fitting method yields a much larger error probability for the Hough transform. In the data given in Figure 4 five out of the eleven data points are outliers, that is, ϵ is approximately 0.45. To achieve 0.01 probability of error in line fitting ($p = 2$), from Table 1 we see that an LMS based line fitting technique should employ only $m = 13$ pairs of points instead of the total 55. Let a be the number of pairs chosen (from the total 15) containing only inlier points. Similarly, let b be the number of chosen pairs (from the total 10) containing only outliers and let c be the number of pairs containing one inlier and one outlier point (a maximum of 30 such pairs). Several triplets of the a, b and c values yield Hough spaces from which the correct line fit cannot be recovered. The probability of such a triplet is

$$\text{Prob}\,(\, a = 3,\ b = 3,\ c = 7 \mid m = 13\,) = \frac{C(30,7)\, C(15,3)\, C(10,3)}{C(55,13)} = 0.0766. \tag{22}$$

Other triplets have similar probabilities and the sum of all the unfavorable cases is much higher than 0.01.

The least median of squares based line fitting method is an application (for $p = 2$) of the general procedure described in Section 3. To obtain the LMS estimates for the parameters β_0 and β_1 first the slopes $\beta_1(j_1, j_2)$ are computed for each pair of points (x_{j_1}, y_{j_1}) and (x_{j_2}, y_{j_2}):

$$\beta_1(j_1, j_2) = \frac{y_{j_1} - y_{j_2}}{x_{j_1} - x_{j_2}} \tag{23}$$

Then for every $\beta_1(j_1, j_2)$ the values

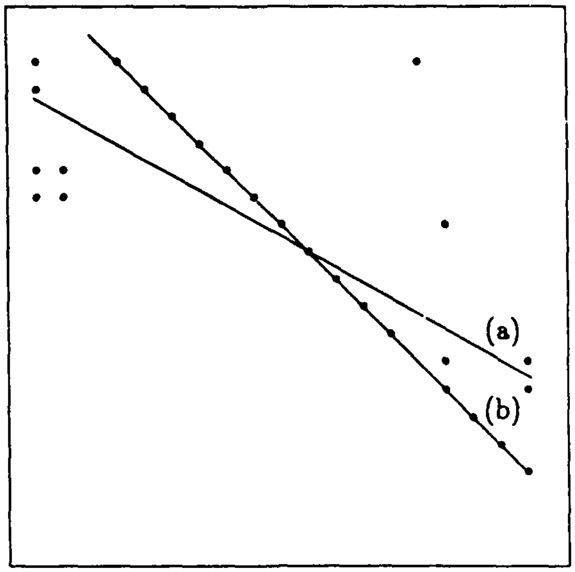


Figure 2. Line fits to data corrupted by symmetric noise. (a) Least squares. (b) MI and LMS algorithms.

$$\alpha_{j_1 j_2}(i) = y_i - \beta_1(j_1,j_2)\, x_i \qquad i = 1, \ldots, n \tag{24}$$

are sorted and the mode of the distribution (10) is computed by the procedure described in Section 3. The mode is taken as the intercept $\beta_0(j_1,j_2)$ and it was shown that the difference (13) is

$$\underset{i}{\text{med}} \mid \beta_0(j_1,j_2) - \alpha_{j_1 j_2}(i) \mid = \underset{i}{\text{med}} \mid \beta_0(j_1,j_2) - y_i + \beta_1(i,j)\, x_i \mid = \underset{i}{\text{med}} \mid r_i \mid \tag{25}$$

The initial values for the parameters of the fitted line are the ones yielding the minimum of (25) for all the m pairs of points considered in the speeded up algorithm.

One-step weighted least squares fit is performed next to increase the relative efficiency of the LMS based procedure to Gaussian noise. The ith data point is given the weight w_i depending on the value of the residual q_i obtained from the initial fit values:

$$w_i = \begin{cases} 1 & \dfrac{\mid q_i \mid}{\hat{\sigma}} \leq 2 \\ \dfrac{3 - \mid q_i \mid}{\hat{\sigma}} & 2 < \dfrac{\mid q_i \mid}{\hat{\sigma}} \leq 3 \\ 0 & 3 < \dfrac{\mid q_i \mid}{\hat{\sigma}} \end{cases} \tag{26}$$

where $\hat{\sigma}$ is the estimated standard deviation of the noise (16). The final estimates of the slope and intercept are the solutions of the classic weighted least squares minimization problem:

$$\min_i \sum_{i=1}^{n} w_i \, [y_i - \beta_1 \, x_i - \beta_0]^2 \tag{27}$$

The algorithm has the LMS part's high breakdown point, but is made more efficient when the underlying residual distribution (i.e. the noise) is Gaussian. The time complexity of the LMS method is significantly reduced by employing the Monte-Carlo type of speed-up technique described in Section 3.

Three different line fitting methods were compared in our experiments: The traditional least squares approach, the median of intercepts (MI) method of Kamgar-Parsi *et al.* (1989), and the LMS based algorithm. In the MI method the intercept and slope is computed for every pair of points the medians of the obtained lists are the MI estimates of the two parameters. As was mentioned at the beginning of Section 2 algorithms in this class have a theoretical breakdown point of 0.293.

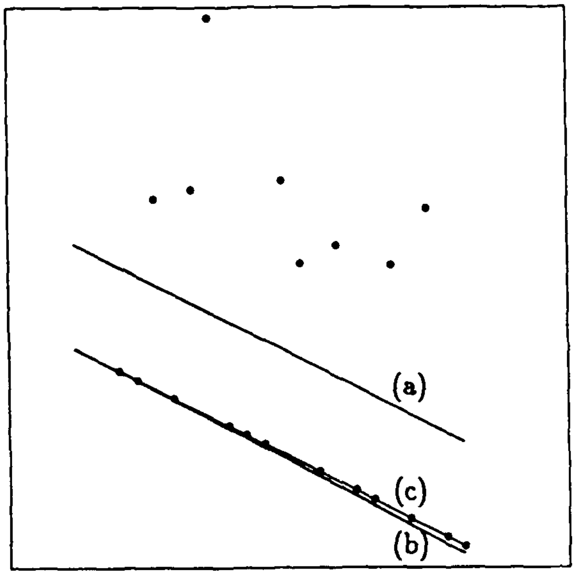


Figure 3. Line fits to data corrupted by asymmetric noise. (a) Least squares. (b) MI algorithm. (c) LMS algorithm.

The data shown in Figure 2 contains fraction $\epsilon = 0.42$ of outliers. While the least squares method fails to find the correct fit (a), the results of the MI and LMS algorithms are identical (b). Although the breakdown point of the median of intercepts method is 0.293, it was able to find the correct line due to the symmetry of the noise distribution.

In the second example (Figure 3) $\epsilon = 0.4$ and the data is corrupted by Weibull distributed asymmetric noise. The Weibull random process that was used had amplitude $A = 2$ and cumulative probability distribution $F(u)=0$ for $u < 0$, $F(u)=1 - e^{-u^2}$ for $u \geq 0$. The least square method (a) fails, the MI algorithm (b) produces a close approximation, and only the LMS algorithm (c) succeeds in finding the original line.

Two different line segments are combined in the third example (Figure 4). Six points belong to one line segment and five to the other, and thus $\epsilon = 0.45$ when we want to fit a line to this discontinuity. Only the LMS method (c) fits the line to the majority of the points.

For the above three examples, the Hough transform correctly finds the line segment corresponding to the majority of the points. However, the Hough transform may fail when systematic errors are present in the data. In Figure 5 a coarsely digitized line segment gave rise to the data points. In this case the three methods recover the correct line, but the Hough transform fails since several false modes are generated by the aligned data points.

We conclude that the robust LMS line fitting algorithm provides the best results for the types of data degradation that were investigated.

6. ROBUST LOCAL OPERATORS

Median and trimmed mean based local operators (L-estimators) have been employed in computer vision for a long time (see for example Bovik *et al.* (1987) for recent results). Recently M-estimators have also become popular. Kashyap and Eom (1988) treated an image as a causal autoregressive model driven by a noise process assumed to be Gaussian with a small percent of the samples (at most 8%) contaminated by impulse noise, i.e. outliers. By employing M-estimators the parameters of the autoregressive process were iteratively refined simultaneously with cleaning the outliers in the noisy image. Besl *et al.* (1988) proposed a hierarchical scheme in which local fits of increasing degrees were obtained by M-estimators. The different fits were compared through a robust fit quality measure to determine the optimal parameters. The authors' claim of a 0.5 breakdown point

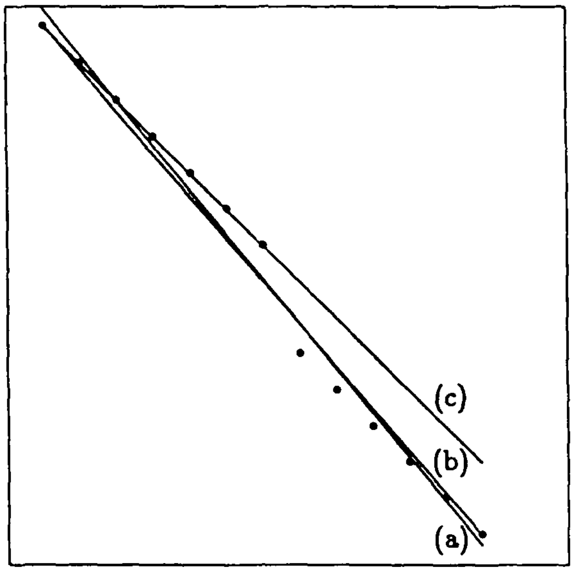


Figure 4. Line fits to a discontinuity. (a) Least squares. (b) MI algorithm. (c) LMS algorithm.

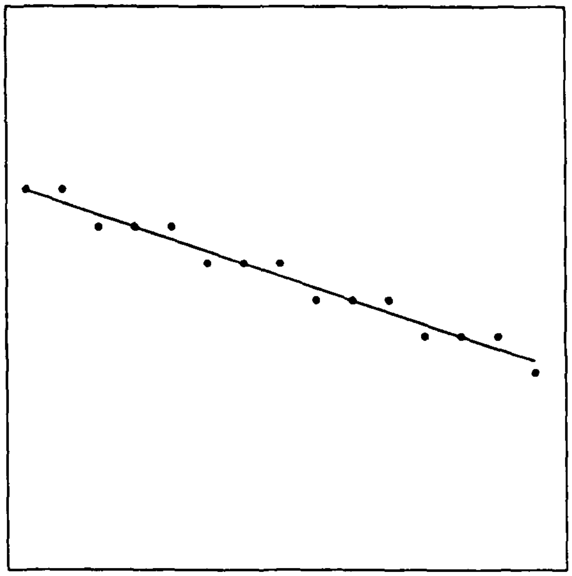


Figure 5. Line fit to quantization effects.

must be regarded with caution since for planar surfaces ($p = 3$) it would exceed the theoretical limit of 0.25 mentioned in Section 2. Haralick and Joo (1988) applied M-estimators to solve the correspondence problem between two sets of 2D perspective projections of model points in 3D. The correct pose solution was then obtained with up to 30% of the pairs mismatched.

The theoretical value of the breakdown point may not be achieved when local operators are applied to image discontinuities. Consider a noiseless, ideal step edge to which a 5 X 5 robust local operator was applied. Assume that 10 pixels in the window belong to the edge (high amplitude) and 15 to the background (low amplitude) and that the center of the window falls on a background pixel. Let the operator have the largest possible breakdown point, 0.5. The operator returns the value of the majority of pixels, that is, the low amplitude of the background.

The image is then corrupted with fraction $\epsilon = 0.2$ of asymmetric noise driving the corrupted samples into saturation at the upper bound. Without loss of generality we can assume that only 3 of the pixels belonging to the background were corrupted in the processing window. There are now 13 pixels with high amplitudes and the operator returns, incorrectly, a high value similar to the amplitude of the edge. Thus, even when the fraction of outliers is much smaller than the theoretical breakdown point of a robust estimator, the operator may systematically fail near transitions between homogeneous regions in images. At transitions, samples of one region are outliers (noise) when fitting a model to the other region and a small fraction of additional noise may reverse the class having the majority.

The size of the local operator also limits ϵ, the maximum amount of tolerated contamination. For a one-dimensional window $2n+1$ pixels long at most n pixels should be corrupted, yielding $\epsilon \le n/(2n+1)$, which only in the limit is 0.5. For example, if $n = 4$ (window length 9) an operator with breakdown point 0.5 tolerates only $\epsilon = 0.44$ contamination.

Least median of squares based local operators perform the algorithm described in Section 5. In every window, the parameters minimizing the median of squares are obtained. Their values can then be employed in various ways. We describe here the smoothing of images corrupted by asymmetric (impulse) noise. This application is of special interest since most smoothing methods fail to achieve good results for this type of noise.

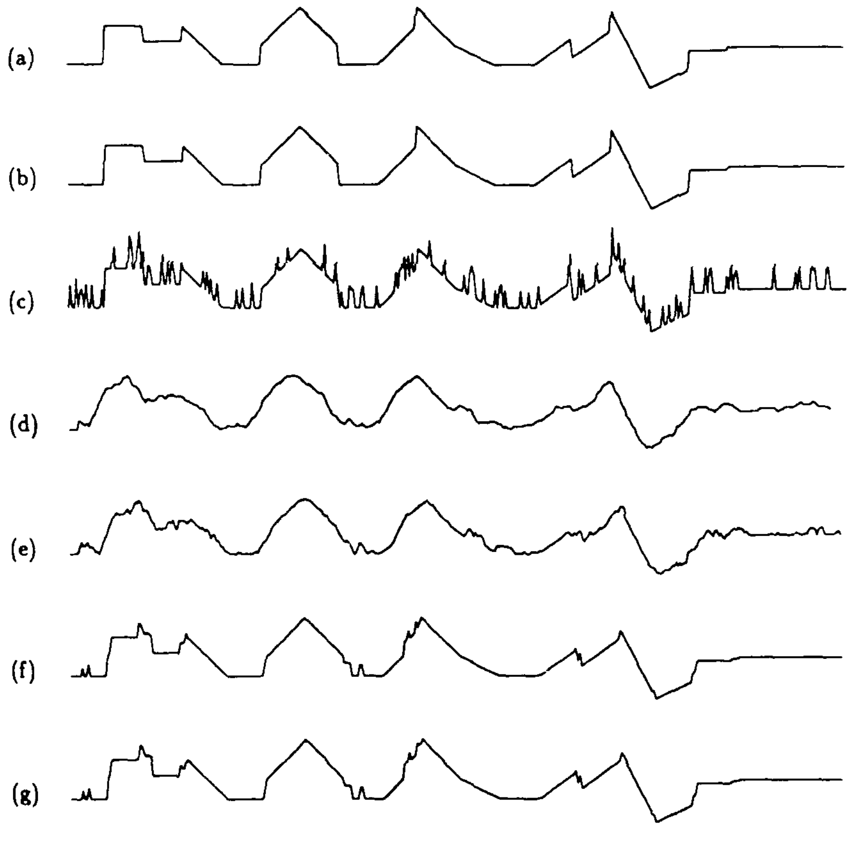


Figure 6. Local line fitting operators applied to one-dimensional data. a) Noiseless data. b) Result of LMS algorithm with speed-up applied to a). c) Noisy data, $\epsilon = 0.22$ fraction of the samples corrupted with Weibull noise. d) Least squares method. e) Result of an M-estimator (Hubel's ρ function). f) LMS algorithm with speed-up. g) LMS algorithm without speed-up.

In a smoothing algorithm the value of the fit in the center of the window becomes the new pixel amplitude. The window coordinates of the center can be taken as (0,0) and thus only the value of the intercept $\hat{\beta}_0$ must be returned by the local operator. Recall, however, that the LMS minimization procedure supplies the values of the other parameters as well.

The experiments were performed with linear (one dimensional) and planar (two dimensional) models. While the algorithm is unchanged for higher order models we have observed that the additional degree of freedom introduced by a second order fit strongly reduces the smoothing achieved in images corrupted by asymmetric impulse noise. If desired, quadratic fits can be applied to the image presmoothed with linear models. It must be mentioned that for images corrupted only with impulse noise, one-step weighted least squares post-processing is not necessary. The LMS algorithm has already eliminated all the noise with possible exceptions around transitions. The post-processing is of importance, however, when Gaussian noise is also present. This case is discussed later in this section.

In Figure 6a a noiseless, piecewise linear waveform containing 400 samples is shown. When an odd sized processing window is applied to the noiseless signal, the majority of the pixels always belong to the region on which the center of window falls. Therefore we have the following important property:
Any noiseless, piecewise polynomial signal built from segments of degree r or less will remain unchanged after being processed by a smoothing operator with 0.5 breakdown point taking into account models of up to order r; in other words, such a signal is a *root* signal of that operator.
Being built from line segments, the waveform remains unchanged when it is processed by the robust LMS smoothing operator of length 9 employing first order models (Figure 6b).

In Figure 6c fraction $\epsilon = 0.22$ of the samples in the waveform were corrupted with Weibull noise of amplitude $A = 8$. The cumulative distribution of this asymmetric noise process was given in Section 5. Note the severe perceptual distortions of the signal around the transition regions. Several smoothing algorithms were then compared using the same processing window size of 9 samples.

When the least squares procedure is applied to obtain the intercept values, the output is oversmoothed simultaneously with the removal of the impulse noise (Figure 6d). A robust M-estimator employing Hubel's ρ function (Hubel, 1981) gave similar results (Figure 6e). To measure the convergence of the estimation process, the Euclidean distance between the points defined by two consecutive estimate pairs $\{\hat{\beta}_0(l), \hat{\beta}_1(l)\}$ and $\{\hat{\beta}_0(l-1), \hat{\beta}_1(l-1)\}$ was employed. The iterations were stopped once this distance became less than 0.05. The poor performance of the M-estimator is caused by the asymmetric nature of the noise and the relative high value of ϵ. While the theoretical breakdown point of this M-estimator is 0.25, the contamination produces regions with much higher ϵ around transitions.

The LMS algorithm with speed-up (Figure 6f) is clearly superior to the previous methods. The original waveform is accurately recovered except at a few transitions where the above discussed artifact appears, and in regions where the contamination exceeds the 0.44 upper bound. Note the recovery of the small step at the right of the waveform. The speed-up is of lesser importance in the one-dimensional case. From the total of 36 possible pairs in the processing window only 19 were considered. Since $\epsilon \leq 0.44$ from Table 1 we obtain the probability of error $Q \leq 0.001$. The probabilistic nature of the speed-up procedure does not degrade the results. The result of the complete LMS algorithm in which all the pairs were considered for the minimization (Figure 6g) does not produce a significantly different result.

In the experiments with two-dimensional data, we have applied the least median square algorithm with speed-up to both synthetic images and natural scenes. The size of the processing window was 5 × 5. Instead of the 2300 possible triplets only 19 were chosen at random, yielding 120-fold speed-up. Since $p = 3$ the assumed contamination is $\epsilon = 0.4$ for a probability of error $Q = 0.01$ (Table 1).

In Figure 7a the perspective plot of a noiseless 64 × 64 synthetic image is shown. The image contains several polyhedral objects and a hollow cylinder. When the LMS smoothing algorithm is applied to the noiseless image (Figure 7b) the only degradation is the removal of the pixels at the corners. This effect is present whenever medians are computed over a rectangular processing window. It can be eliminated by selectively computing the medians along principal directions (0, 45, 90 degrees).

In the noisy image fraction $\epsilon = 0.15$ of the samples were corrupted with a Weibull random process having amplitude $A = 75$ (Figure 7c). The output of the LMS smoothing algorithm is given in Figure 7d. Note that while the noise is completely cleaned in the uniform regions distortions may remain around transitions. The processing took 385 seconds of CPU time on a VAX 11 785 computer. Using parallel hardware instead of a serial machine much faster processing times could be achieved.

A 128 × 128 aerial scene (upper left, Figure 8) was also corrupted with Weibull noise having $A = 255$ (upper right). The employed noise process was equivalent to removing $\epsilon = 0.18$ of the pixels and replacing them

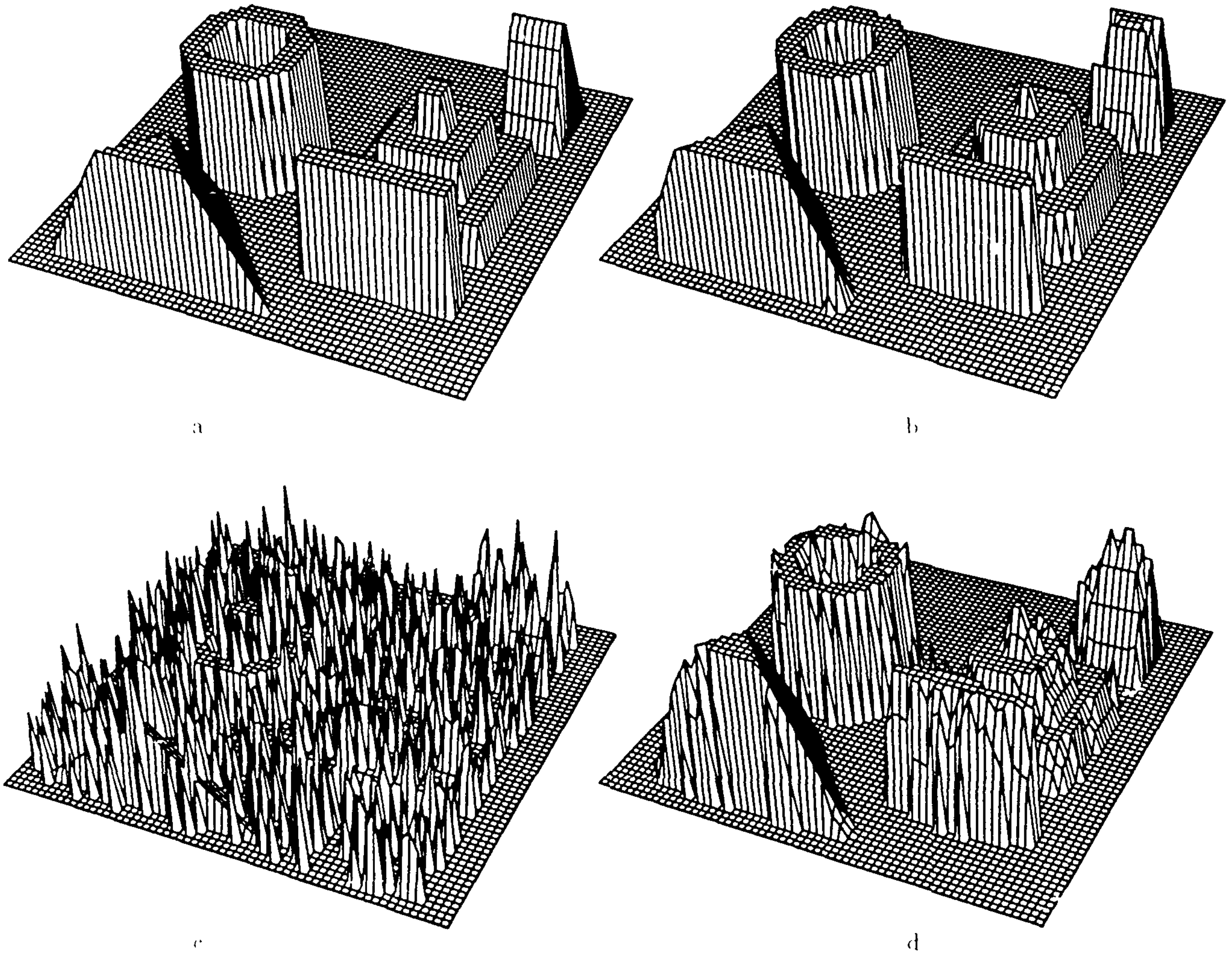


Figure 7. Perspective plots of a synthetic image. a) Noiseless. a) After application of the LMS algorithm. b) Corrupted with Weibull noise. c) After application of the LMS algorithm.

with the maximum gray level value. The image smoothed by the LMS algorithm is shown at the lower left. Most of the details are recovered and thus the algorithm produced a nonlinear interpolation of the missing samples. The processing took 1623 seconds. The result obtained by employing a 5 × 5 Gaussian weighted smoothing window is given for comparison at the lower right of Figure 8. A serious amount of blurring is introduced, showing the superiority of the robust method for impulse noise removal.

The importance of the one-step weighted least squares post-processing can be observed from Figure 9. In Figure 9a the waveform of Figure 6a is shown corrupted with zero mean Gaussian white noise, standard deviation $\sigma = 2$. Again all the smoothing operators used window size 9. The result of the least squares algorithm is given in Figure 9b. The least squares method is optimum for homogeneous regions and in those regions the results can be regarded as the bound on the achievable performance for the given window size. (Recall that the least squares method fails at transitions, Figare 4.) The result of Hubel's M-estimator (Figure 9c) is similar to the least squares output.

When the LMS algorithm followed by a one-step weighted least squares procedure is employed (Figure 9d) an improvement in the recovery of transitions (edges) and an increase in the noise related fluctuations in the homogeneous regions can be observed. These fluctuations, however, were already attenuated by the post-processing, as a comparison with the output of the LMS algorithm applied alone (Figure 9e) shows.

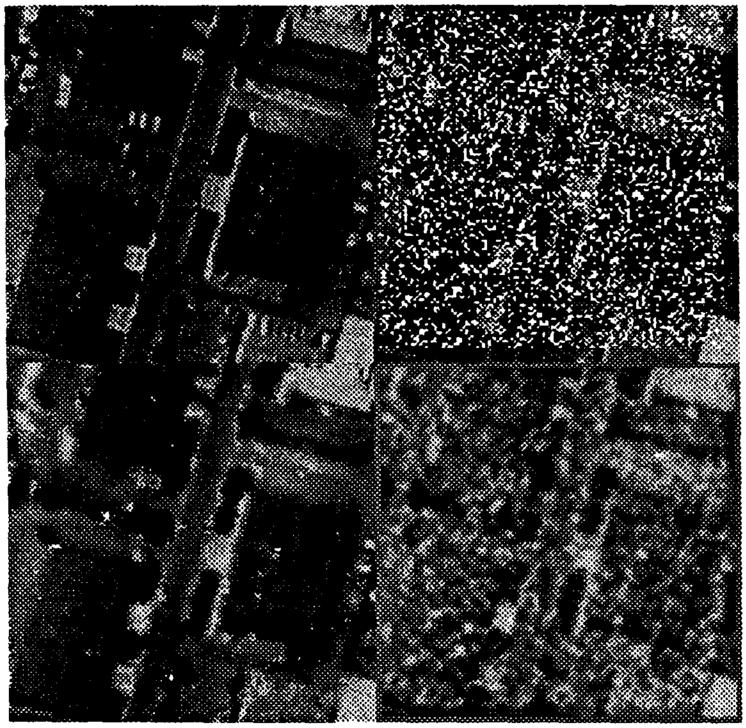


Figure 8. Aerial image. Upper left: Noiseless. Upper right: Noisy with 15% of the the samples removed. Lower left: After application of the LMS algorithm. Lower right: Gaussian smoothing.

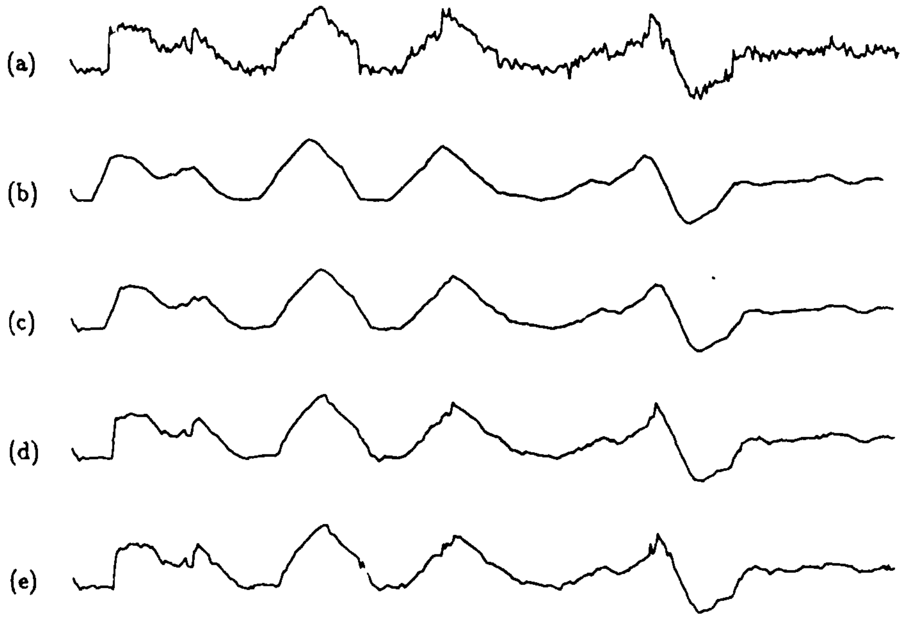


Figure 9. Smoothing of waveform corrupted by Gaussian noise. a) Noisy data, $\sigma = 2$. b) Least squares method. c) Result of Hubel's M-estimator. d) LMS algorithm and one-step weighted least squares. e) LMS algorithm alone.

7. FURTHER DIRECTIONS OF RESEARCH

We have presented a novel robust paradigm based on the least median of squares method for solving computer vision problems. Severa' directions for further investigation are suggested:

- Design of robust clustering algorithms in which the number of clusters is not known a priori.
- Development of a Hough transform variant in which the analysis of the accumulator is done by the clustering technique described in Section 4. The dependence of the Hough transform's accuracy on the chosen bin size and the effect of digitization of the data will be eliminated.
- Employing heuristics to improve the performance of LMS based algorithms around transitions between homogeneous regions in images. If local connectivity can be established *before* processing the impulse noise can be separated from the samples belonging to the adjacent regions. Successive application of differently sized windows (a multiresolution approach) may also be employed to dichotomize a local region into data and impulse noise.
- To improve the performance of the LMS algorithm's output in the presence of zero mean, symmetric noise processes (e.g. Gaussian), the following approach may be helpful. The input is first smoothed by an LMS procedure which better preserves the transitions. The resulting signal is presegmented into homogeneous regions which then are processed with a robust M-estimator.
- The capacity of the LMS smoothing algorithm to act as a nonlinear interpolation scheme, which preserves transitions better than do linear methods, can be employed in solving computer vision problems where irregularly sampled data is frequent (stereo, optical flow etc.).
- Development of LMS based algorithms for other computer vision problems in which regression analysis is involved.

References

[1] D.F. Andrews (1974): A robust method for multiple linear regression. *Technometrics*, **16,** 523–531.

[2] A.E. Beaton and J.W. Tukey (1974): The fitting of power series, meaning polynomials, illustrated on band-spectroscopic data. *Technometrics*, **16,** 147–185.

[3] P.J. Besl, J.B. Birch and L.T. Watson (1988): Robust window operators. *Proceedings of the Second International Conference on Computer Vision*, December 1988, Tampa, Florida, 591–600.

[4] A.C. Bovik, T.S. Huang and D.C. Munson, Jr. (1987): The effect of median filtering on edge estimation and detection. *IEEE Trans. Pattern Analysis Machine Intelligence* **PAMI-9,** 181–194.

[5] G.W. Brown and A.M. Mood (1951). On median tests for linear hypotheses. *Proceedings of the 2nd Berkeley Symposium on Mathematical Statistics and Probability*, J. Neyman (Ed.), University of California Press, Berkeley and Los Angeles, 159–166.

[6] K.S. Cheng and T.P. Hettmansperger (1983): Weighted least-squares rank estimates. *Commun. Stat.* **A12,** 1069–1086.

[7] R.M. Haralick and H. Joo (1988): 2D-3D pose estimation. *Proceedings of the 9th International Conference on Pattern Recognition*, November 1988, Rome, Italy, 385–391.

[8] S. Heiler (1981): Robust estimates in linear regression—A simulation approach. In *Computational Statistics* H. Büning and P. Naeve (Eds.), De Gruyter, Berlin, 115–136.

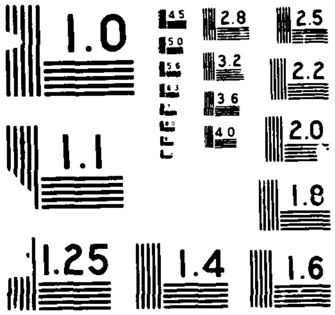
[9] P.W. Holland and R.E. Welsch (1977): Robust regression using iteratively reweighted least squares. *Commun. Stat.* **A6,** 813–828.

[10] P.J. Huber (1981): *Robust Statistics*. John Wiley & Sons, N.Y.

[11] L.A. Jaeckel (1972): Estimating regression coefficients by minimizing the dispersion of residuals. *Ann. Math. Stat.* **5,** 1449–1458.

[12] A.K. Jain and R.C. Dubes (1988): *Algorithms for Clustering Data*, Prentice-Hall, Englewood Cliffs, NJ.

AD-A212 981   PROCEEDINGS OF IMAGE UNDERSTANDING WORKSHOP HELD IN   13/13
PALO ALTO CALIFORNIA O. (U) SCIENCE APPLICATIONS
INTERNATIONAL CORP MCLEAN VA L S BAUMAN MAY 89
UNCLASSIFIED N00014-86-C-0700   F/G 17/11   NL


1.0
45
50
56
63
2.8
3.2
36
40
2.5
2.2
2.0
1.1
1.8
1.25
1.4
1.6

[13] J. Jolion and A. Rosenfeld (1988): Cluster Detection in Background Noise, CAR-TR-363, Center for Automation Research, University of Maryland, College Park, June 1988. To appear in *Pattern Recognition Letters*.

[14] I.M. Johnstone and P.F. Velleman (1985): The resistant line and related regression methods. *J. Am. Stat. Assoc.* **80,** 1041-1059.

[15] B. Kamgar-Parsi, B. Kamgar-Parsi and N.S. Netanyahu (1989): A nonparametric method for fitting a straight line to a noisy image. To appear in *IEEE Trans. Pattern Analysis Mach. Intell.* See also CAR-TR-315, Center for Automation Research, University of Maryland, College Park.

[16] R.L. Kashyap and Kie-Bum Eom (1988): Robust image models and their applications. In *Advances in Electronics and Electron Physics, Vol. 70*, Academic Press, San Diego, CA, 79-157.

[17] G. Li (1985): Robust regression. In D.C. Hoaglin, F. Mosteller and J.W. Tukey (Eds.), *Exploring Data Tables, Trends and Shapes.* John Wiley & Sons, 281-343.

[18] F. Mosteller and J.W. Tukey (1977): *Data Analysis and Regression.* Addison-Wesley, Reading, MA.

[19] P.J. Rousseeuw (1984): Least median of squares regression. *J. Amer. Stat. Assoc.* **79,** 871-880.

[20] P.J. Rousseeuw and A.M. Leroy (1987): *Robust Regression & Outlier Detection.* John Wiley & Sons.

[21] A.F. Siegel (1982): Robust regression using repeated medians. *Biometrika* **69,** 242-244.

[22] J.M. Steele and W.L. Steiger (1986): Algorithms and complexity for least median of squares regression. *Discrete Applied Math.* **14,** 93-100.

[23] H. Theil (1950): A rank-invariant method of linear and polynomial regression analysis (parts 1-3). *Ned. Akad. Wetensch. Proc. Series A* **53,** 386-392, 521-525, 1397-1412.

[24] V.J. Yohai and R.H. Zamar (1988): High breakdown point estimates of regression by means of the minimization of an efficient scale. *J. Amer. Stat. Assoc.* **83,** 406-413.

Perceptual Grouping of Curved Lines[1]

John Dolan Richard Weiss

Dept. of Computer and Information Science
University of Massachusetts
Amherst, MA 01003

Abstract

Image curves often correspond to the bounding contours of objects as they appear in the image. As such, they provide important structural information which may be exploited in matching and recognition tasks. However, these curves often do not appear as coherent events in the image; they must, therefore, be (re)constructed prior to their effective use by higher-level processes. The system described herein exploits principles of perceptual organization such as proximity and good continuation to identify co-curving or curvilinear structure. This process entails three subprocesses: linking, grouping, and replacement. Components of each structure are first organized based on pairwise geometric consistencies. Groups of these linked components are then classified according to the geometric trend apparent at a single perceptual scale. Finally the constituents of each group are replaced by a single curve, thus making their coherence explicit. The system is iterative, operating over a range of perceptual scales—fine to coarse—and yielding a hierarchy of alternative descriptions. Results are presented for two images showing the performance of the system using only straight-line and conic replacement models and indicating the reasonableness of the paradigm for more complex replacements such as inflections, corners, etc.

1 Introduction

Image curves often correspond to the bounding contours of objects as they appear in the image. As such, they provide important structural information which may be exploited in matching and recognition tasks. There is general agreement about the importance of extracting and describing image curves, but there seems to be no consensus about how best to do this. More importantly, agreement is lacking about which are the salient features of curves and, therefore, which features are worthy of description [2,11,16]. Part of the problem, as Fischler and Bolles have observed [8], is that *perceptual significance* is directly related to the goals of the system. Thus, for a parts inspection system with fixed viewpoint, a faithful tracing of the edge contour might suffice. On the other hand, our goal is to facilitate the recognition of 3D objects in terms of image curves so, as suggested by Lowe [15], only those features that remain invariant or quasi-invariant over a wide range of viewpoints are perceptually significant. This means that 2D curve features such as collinearities, inflections, smooth curves, corners, and cusps are important to explicitly detect and describe.

With respect to image curves, a general definition of *perceptual grouping* can thus be formulated as: the search for and explicit description of *significant curvilinear structure*—i.e., any of the invariant curve features mentioned above. To this end, we have developed a computational framework, much like that of Boldt and Weiss [3], that measures geometric relations on symbolic/geometric entities called *tokens*. This geometric analysis is guided by principles of *perceptual organization*: ideas originating with the Gestalt psychologists and discussed, more recently, by Kanizsa [13], Lowe [14,15], and others [3,24,26,8,23]. Within the system we are currently building, the perceptual organization of image curves consists of the successive, hierarchical organization of tokens into coherent curvilinear events.

The remainder of this section discusses: problems associated with digital curves; the role of scale; our choice of primitive descriptors; and computational issues. The treatments will be brief, intending to acquaint the reader with the overall rationale of our design choices, rather than provide an in-depth treatment of each issue.

1.1 Problems with digital curves

In the context of image curves, the goal is the discovery and explicit description of significant curvilinear structure. However, two problems, associated with the appearance of curves in the digital image, prevent a straight-forward approach. First, curves are not explicitly present in the image; rather, they are encoded as differences in intensities of neighboring pixels. This means that some detection process, often like *edge detection* [2] based on local differencing, is required to make local portions of a curve explicit. Second, the continuity and coherence of image curves is destroyed by a combination of factors including: aliasing due to the geometry of the sample grid; digitization error associated with the sampling method; noise introduced during imaging; as well as "natural phenomena" in the scene itself—such as occlusion and shadows. Moreover, because of its simple local nature, the detection process itself can add to the difficulties. In fact, any detection process can and will result in: *gaps*—false negatives—where subtle

[1] This work was supported by the Defense Advanced Research Projects Agency under grant F30602-87-C-0140 and by the National Science Foundation under grant DCR-8500332.

[2] A process, like region segementation, that is based on similarity of features might also be used.

but significant structures are missed; *spurious locations* false positives—where events that do not correspond to significant structure are marked by the detection process; *deviations in position and uncertainties of orientation*—where the detection operation is ill-defined (e.g., corners) or where local estimates of these quantities are adversely affected by noise. This suggests that some organizing process, like *perceptual grouping*, is required to overcome the effects of such fragmentation/errors—i.e., to restore the coherence of image curves.

1.2 Scale dependence of structure

Structure and its description vary with *perceptual scale*. Here perceptual scale is equivalent to spatial extent [3,14]—corresponding in some sense to the radius of activity of visual cells [22]—as distinguished from "scale" in the sense of scale-space [24,17], which corresponds to a frequency decomposition of the signal. As a simple example of perceptual scale, consider the curving contour shown in Figure 1. At a very fine scale, say at the level of the edge-detection process, a small portion of the contour in Figure 1a may appear as a straight segment. At the somewhat larger scale of Figure 1b, simple curving structure is apparent. At the still larger scale of Figure 1c, complicated curving structure is perceived. Finally, at a very large scale, Figure 1d, the contour again appears straight.

It is therefore reasonable to conclude as others have [14,23,25] that structure can and does occur over arbitrary spatial extent—in Figure 1, straight structure was apparent at both the finest and the coarsest perceptual scales. Thus, it is necessary to search for significant structure at various scales and to create explicit descriptions of the structure found at each of those scales. Furthermore, a hierarchical description is desirable, since the structural description of any one scale is not likely to be valid across all scales.

1.3 Primitive descriptors

The vocabulary of primitive descriptors must be sufficiently rich to describe all significant structures of interest: collinearities, smooth curves, inflections, corners, cusps. In addition, primitive descriptors should be computationally tractable—i.e., reliably and efficiently computed from few data. We have chosen *straight-line* and *conic-spline* descriptors by the following rationale.

Collinearity is invariant under all but degenerate views. Its presence in the image is thus highly significant, and it therefore merits an explicit primitive descriptor—the straight line. The usual choices to describe smooth curves are conics [21,18,1] and cubics [6.20]. Conics have fewer parameters and provide greater simplification, but cubics are arguably richer in descriptive power. However, inflections are also highly invariant and therefore should be explicitly recognized and desciibed. As a composition of conics, inflections must be *explicitly* represented—because no conic can cross the inflection point. Cubics, on the other hand, render inflections implicit if knots are chosen at curvature extrema. If knots are placed at inflections, there is little justification for the additional cost/complexity of cubics. Thus we choose conics, and because we will want to impose continuity constraints across inflections, we have chosen the spline form of conics. Finally, corners and cusps are most naturally handled as assemblages of primitive descriptors, since it is desirable to explicitly note the location of these discontinuities.

1.4 Computational complexity

Structure is an n'ary (as opposed to binary) geometric relation—i.e., it is defined over sets (rather than pairs) of tokens. Searching for a significant curvilinear structure (e.g., inflections) is thus equivalent to determining a single such n'ary relation on the set of all tokens, which in itself is combinatorially explosive. As was shown by Boldt and Weiss in the case of straight lines [3], this amounts to examining all subsets of cardinality at most n—for a set of m tokens this is $\sum_{i=2}^{n} C(m,n)$ which is $O(m^n)$. In the case of curvilinear grouping, the search is for all relations of interest for curves (inflections, corners, etc.) and at multiple scales; this is clearly blows up to an impossibly large task unless this search is somehow constrained.

We propose to manage this complexity by the techniques of Linking, Grouping, Replacement, and Iteration.

- Linking. Our contention is that significant curve structure exhibits simple binary consistencies[3] (e.g., proximity, angular compatibility, etc.) among neighboring components. Thus, simple binary tests can be used to build a graph of *feasible sequences*. Then, in searching for structural relations, it is no longer necessary to consider all subsets of tokens, but only sets of feasible-sequences.
- Grouping. A set of simple geometric tests is used to determine which structural relations, if any, are applicable to a given sequence. Only sequences corresponding to significant structure are passed to the replacement module. By limiting possible replacements, the overall cost of constructing replacements is minimized and the appropriateness of the replacements is better assured.

[3] This view has been advanced in the case of straight lines by Boldt and Weiss [3] and in the case of conics by Pavlidis [21].

- Replacement. Each sequence of tokens is replaced by a single token. Provided redundancy among sequences can be kept low, a dramatic reduction in the total number of tokens from level to level can be achieved.
- Iteration. Each iteration consists of linking, grouping, and replacement. Iterating over a sequence of scales (fine to coarse) means it is necessary to search only one scale at a time, with linking and grouping controlling the complexity within that scale. Replacement manages complexity across scales: as the spatial area to be searched increases, the total number of tokens in that area should remain constant or, ideally, decrease due to replacement.

It makes sense to iterate fine to coarse, since the initial tokens are the result of a highly-local detection process. Furthermore, such iteration facilitates the construction of the desired hierarchical descriptions.

2 Computational Paradigm

In the previous section it was argued that an exhaustive search for an optimal correspondence between local contour fragments and models of significant structure is not feasible, even for a single such model. By contrast, the system described herein applies geometric rules hierarchically to solve a local search problem. *Linking* finds, within the set of tokens, pairs satisfying the binary relations of the particular grouping principles employed. *Grouping* performs a detailed geometric analysis on sets of linked tokens whose extent is limited by the current perceptual scale. This analysis entails a classification of each token sequence according to its geometric configuration and an evaluation reflecting how good it is within its class and across classes. Only the "n" best sequences are retained for the replacement process.[4] *Replacement* encodes the geometry of a surviving group by substituting a single primitive descriptor or an assemblage of such descriptors. The key rationale for each of these steps is simplification of computation. The linking step filters out most combinations of tokens which are not likely to lie on the same curve. Grouping simplifies the replacement step since only one type of fitting procedure needs to be applied. Replacement reduces the number or complexity of tokens to be considered by the next linking phase, since a sequence of tokens is replaced by a single new token or a simpler sequence of new tokens.

Scale, as indicated previously, is a key issue in perceptual grouping both for computational and descriptive reasons. The processes of linking and grouping cannot be applied to the entire image at once; it is necessary to restrict the number of combinations which are examined by each of these processes. In the case of linking, this is done by examining only those tokens that are within a small distance from each other. In the case of grouping, the combined lengths of tokens is restricted. This latter distance constraint is referred to as the *perceptual window*. Since the replacement process reduces the number of tokens, the perceptual window can be enlarged in successive cycles. Scale is also important for the description process. In particular, curvature depends on scale, and this will influence decisions whether a line is straight or curved and whether a curve is smooth or has a corner.

2.1 Initial Token Generation

The goal of the initial token generation process is to produce a set of tokens that are approximations to the local edge structure. Finding an optimal solution to this problem has been studied by many researchers [5,4,9] and is not considered here. The output of this process, a field of unit tangents, is used by subsequent processes to construct more-comprehensive/less-local descriptions of the underlying curve events. Currently, we employ a two-stage process that is virtually identical to the one used for straight lines by Boldt and Weiss [3]. First, edges are localized by finding zero-crossings of the Laplacian; then, local orientation is estimated by computing the gradient at each such edge point.[5]

The edge direction for every identified edge point (e_i) is defined to be perpendicular to the gradient at that point and is given by the vector $(-I_y(e_i), I_x(e_i))$. For each edge point, an edge token is generated as the unit line segment in the edge direction and centered at the edge location. Each token is thus a local approximation of the position and orientation of the edge structure and, as such, it represents a "best" local description of that structure. The orientation defines a direction of the token, which is an ordering of the endpoints. For a more complete discussion of this entire process see [3] and also [7].

The principles of grouping employed by subsequent operations of the system are independent of the processes that generate the initial set of tokens, although in general the results will be different. Thus, the system can be applied to tokens produced by other edge detectors or by processes other than edge detection (e.g., region segmentation).

[4] Ungrouped tokens are carried forward to the next iteration if they are still active—i.e., if they were created by a recent enough iteration.

[5] We have recently coded a version of the Canny's edge-algorithm [4] for performance comparison.

2.2 The Linking Process

In order to recover curvilinear structures, it is first necessary to identify which fragments form the same curve; or more precisely, which tokens are potentially neighbors along the same curve. For tokens to be neighbors they will, in general, satisfy constraints on certain geometric measures, especially in terms of *proximity*, *angular compatibility*, and *continuation*.[6] See Figure 2. The constraints imposed on these measures by each of the structural models (inflection, corner, smooth curve, etc.) help prune the search space so that the grouping process need only consider feasible sequences of tokens rather than all subsets of tokens. However, care must be taken in applying these constraints so that the probability of rejecting correct links and the probability of accepting incorrect links are both sufficiently low.

From a graph theoretic point of view, the task of linking is one of constructing the graph of tokens under the following compatibility relation: $\mathcal{R}(T) = \{(t_i, t_j) | t_i, t_j \in T \wedge \delta(t_i, t_j) < \rho \wedge |\theta_i - \theta_j| < \theta \wedge t_i \parallel t_j < \lambda\}$; where T is the set of tokens (nodes), $\delta(t_i, t_j)$ is the minimum distance between the endpoint of token t_i and any point of token t_j (the link radius), $|\theta_i - \theta_j|$ is the minimum angular difference between the endpoint tangents of tokens t_i and t_j(the link angle), and $t_i \parallel t_j$ is the percentage of overlap of token t_i and token t_j, projected onto t_i (the link overlap). The maximum allowable values for link radius, link angle, and link overlap are given by ρ, θ, and λ, respectively. Note, the overlap criterion embodies the notion of continuation, in that for a token to represent the continuation of a contour we require that the percentage of its length that overlaps a neighboring token be sufficiently small. Put another way, to represent the continuation of a contour, a token should extend either forward or backward (in the sense of its directedness) from a neighboring token a sufficient portion of its own length.

2.3 The Grouping Process

The grouping process is concerned with analyzing the geometric structure of sets of tokens which, after the linking process, are believed to constitute contiguous pieces of curvilinear contours. The analysis, in a sense, matches pieces of contour to models of *straight line, conic, inflection, corner*, and *cusp*. Taking the results of the linking phase as a compatibility graph over the set of tokens, this entails enumerating for each token all paths of the subgraph contained within a perceptual window centered at the token, where the size of the window is given by a parameter (perceptual radius) that corresponds to the perceptual scale of the particular level of iteration. Each such concatenation of tokens we term a *strand*, and it consists of three parts: a backward path into the initial endpoint of the token, the token itself, and a forward path from the final endpoint. Each strand is then classified and evaluated in terms of the geometric properties that it exhibits. The "best" n strands (where n is some small integer) for each token are passed to the replacement module. Thus, the purpose of grouping is to identify those strands at a given scale that will yield, in terms of the set of primitive descriptors, the clearest and most appropriate description of the contour events passing through each token. We will not discuss issues of path enumeration here other than to say that such enumeration may be treated as a graph traversal problem, with neighborhoods of the graph expanded via search trees. Neither will evaluation be discussed here. A more complete treatment is available in [7]; the present discussion will focus instead on classification and issues of redundancy.

The purpose of classification is to explicitly identify which significant curvilinear structure obtains for each strand at the current perceptual scale so that it may be appropriately and explicitly represented by the replacement step. The classes ot curvilinear events that we discriminate are: STRAIGHT, CONIC, INFLECTION, CORNER, CUSP, and UNKNOWN. Moreover, classification mediates between complex curvilinear events and the set of primitive descriptors (straight lines and conics). This is useful because of the relatively high cost of directly fitting each of our primitive descriptors, which would be necessary if the type of event were unknown. In our case, this would involve at least straight-line and general-quadratic fits. More significantly, there are many commonly occurring curvilinear events, like inflections and corners, which involve combinations of straight lines and conics. In such cases, direct fitting would be inappropriate and uninformative, whereas, the explicit information obtained in the classification step can be used to select a descriptor that better agrees with perception. Figure 3 shows an example in which minimizing the fit error alone does not produce an adequate description.

Classification is accomplished by measuring the positional and angular distributions of the tokens comprising a strand. Since the tokens satisfy the geometric constraints of the linking process, they are hypothesized to lie on a single curve structure, and a polygonal approximation to this is constructed by connecting sample points of the strand as shown in Figure 4. This really amounts to a reparameterization of the tokens in terms of points. So, for example, a straight line is fully specified by its endpoints; so too a conic is fully specified by any 5 points on the curve. The polygonal approximation is closed by a chord connecting the first and last sample points. Now, if the

[6] Other measures such as contrast across tokens, intensities of abutting regions, etc., could be used to further refine the determination of neighbors.

"aspect ratio" (width/length) of the resulting polygon is sufficiently small, the strand is classified as STRAIGHT—i.e., the strand fits within a narrow bounding rectangle.[7] The positional and angular constraints on a polygon inscribed in a conic are discussed in detail by Pavlidis [21], and by these we are able to discriminate CONIC-like strands. These same measurements—like the angular differential between adjacent tokens—are also useful in discriminating the remaining classes. For example, a strand that crosses its chord while maintaining a smooth angular differential passing through zero is considered an INFLECTION. A strand lying entirely on one side of its chord, exhibiting an angular discontinuity and a sign change in the angular differential, is classified a CUSP; likewise, a similar strand with no sign change in the angular differential is a CORNER. Any strand classified as one of these significant curve structures is termed *grouped*. Finally, a strand not fitting any of these categories is called UNKNOWN. This only means that, at the current scale, the strand is beyond the descriptive power of our primitive descriptors.

One of the consequences of constraining the grouping process to perceptual windows centered at each token is that a token will often be a member of more than one strand. In the instances where these strands represent alternative descriptions- i.e., they describe different curve events—their retention is desirable. However, multiple strands may also reflect redundancy in the grouping process—i.e., single pieces of the underlying structure may be "discovered" multiple times. Such redundancies are obviously detrimental to system performance: actually increasing rather than decreasing the number of tokens. Operationally there are two types of redundancy. The first type occurs when one strand is a sub-sequence of another. This type is relatively easy to detect and eliminate directly by a process that amounts to selecting strands of maximum cover. The second type occurs when neighboring strands share significant overlap, but neither strand is entirely subsumed by the other—i.e., each strand accounts for slightly different portions of the data. This type gives rise to a sort of feathering of tokens, especially on curving contours (as may be seen in the second experimental example). Since there is no strand of maximum cover, redundancies of this type should properly to be treated as collateral grouping, where neighboring quasi-parallel strands are bound together for a single replacement.

2.4 The Replacement Process

The primary function of the replacement process is to generate a description for every strand of tokens (after elimination of redundancies) that has been grouped during the current iteration. For each strand, this amounts to constructing a best-fit replacement token of the appropriate class and updating the database of tokens with the resulting token. In addition, any tokens that have not been grouped but that are still current[8] remain active. Tokens that have not been successfully grouped and are no longer current are simply dropped. The generation of new tokens entails three basic operations: 1)*description* (constructing the"curve" that best summarizes the geometry of the configuration of tokens); 2)*feature determination* (the calculation of token attributes, like contrast, aggregate error, etc. in general by measurements either directly on the tokens themselves or on their constituents); and 3)*graph construction* (the maintenance of internal pointer linkages—e.g., between a replacement token and its constituents—yielding the hierarchical aspect of description).

The discussion that follows will focus primarily on issues of description; the feature determination and graph construction tasks are treated, in detail, elsewhere [7]. We distinguish two types of replacement, *simple replacement* and *complex replacement*, depending on the correspondence between the particular curve event and the primitive descriptors. Straight lines and conics entail simple replacement since there is a unique primitive available for each. However, the other classes (inflections, corners and cusps) each entail complex replacement since their proper description requires an assemblage of primitive descriptors. Figures 5a and 5b show examples of simple replacement, while 5c and 5d show examples of complex replacements. Figure 5a shows the simple replacement of a sequence of straight tokens by a single conic spline. Figure 5b illustrates the simple replacement of a sequence of inflected curves and a straight line by single straight line. Figure 5c illustrates the how complex replacements are for corners, inflections, and cusps are handled by *point-event tokens*. These are effectively place-tokens that act as concatenation operators on the primitive descriptors. In Figure 5d, a more general point-event token the SMOOTH-JOIN is shown. This allows neighboring sequences to joined and simplified even if there is no corresponding curve-event class. Also shown in 5d, the recursive nature of these complex replacements permits the composition of arbitrarily complicated curves from two simple primitives.

If the class of a token group is STRAIGHT, a least squares principal axis method is used to fit the appropriate line segment. If the class is CONIC, then we first derive the endpoint tangent conditions yielding a one-parameter family of conic sections. From this, we select the one that minimizes the total point-to-curve distance over the tokens of the strand. If the total angle, through which the conic turns, exceeds a threshold ($\approx \pi$),[9] then the strand is

[7] In fact, Lowe [14] rightly points out that a quasi-uniform distribution within a narrow rectangle is a better indicator of linearity.

[8] A current token is one created by a recent iteration.

[9] The rational parametric form of the spline is unstable at π, and the desirable convex hull property is violated for angles $> \pi$.

subdivided and each resulting subdivision is fit with a conic—successive conics are constrained to $G^{(1)}$ (tangent) continuity. Similarly, an INFLECTION is subdivided at the predicted inflection point into two conics whose curvature is oppositely signed but constrained to $G^{(1)}$ continuity at the inflection. Strands of CORNER and CUSP classes are also subdivided at the predicted location of the critical event, and the components are appropriately fit with only a positional or $G^{(0)}$ continuity constraint. Strands classified as UNKNOWN are not considered grouped and are therefore not replaced. They are either carried forward to the next iteration or dropped.

3 Preliminary Results

Preliminary experimental results are shown for two images: an image of a chimney from an outdoor scene of a house, and a near-binary image of a calligraphic symbol: the letter "m". These images were chosen to demonstrate performance on essentially straight data (chimney) and extended, simply curving data (Big M). They were also selected because the significant structure conforms roughly to the perceptual scale of the early iterations. Redundancies of the first type (cf. section 2.3) have been subsumed—i.e., only strands of maximum cover are retained. Only simple replacements are shown—i.e., straight-lines and conics. Replacement by aggregate descriptors (corners, cusps, and inflections) is coded, but not yet fully operational. Identical initial settings of the system's parameters were used to produce all results. The link-angle was 60.0°, the link-overlap was 50%, the link-radius was 0.25 pixel-units, and the perceptual radius (1/2 the maximum strand length) was 2.5 pixel-units. The radii were scaled by a factor of 1.5 per iteration. An argument could be made for different settings—in particular, the link-angle may seem overly generous. However, in the early iterations, the system should be forgiving of perturbations resulting from digitization, noise, etc.; and, in practice, these settings have given satisfactory results.

Chimney. This example is meant to illustrate the performance of the system in the face of essentially straight data. The sytem has done a reasonable job on straight data in spite of the more complicated machinery required to handle more general curving structures. Figure 6a shows the original gray-scale sub-image, which is a (16 × 16) detail of the chimney of a house, taken from an original (256 × 256) outdoor scene. Figure 6b shows the initial edge tokens extracted from this sub-image. In Figure 6c the results of the first replacement cycle (representing structure 4–5 pixels in length) are shown. Already the major vertical edges of the chimney are reconciled as nearly vertical straight lines. The left top diagonal of the chimney is also a single line. Each of these lines represents a reduction of roughly 5 tokens to a single token. By contrast, the right top diagonal is a collection of conics and overlapping straight-lines, the result of redundancy of the second type (cf. section 2.3) and alternative description.[10] Figures 6d, 6e, and 6f show the results of the second, third, and fourth iterations respectively. It is interesting to trace the evolution of the two major diagonal lines of the roof. The lower diagonal starts as fairly smooth curving arrangement of lines and, largely because of type-two redundancy, evolves into a band of feathered lines and conics. In fact, this band becomes progressively more dense, indicating the combinatorial nature of redundancies. Compare this with the upper diagonal, which starts as a rather noisey arrangment of unit tangents and evolves into a single straight line. Ironically, the initial noise here suppresses type-two redundancies; and those of type-one are eliminated by the system. Thus, an appropriate and simple description is derived, albeit fortuitously.

Statistics for the first four iterations are as follows. Iteration 1: 170 links created; 89 straight lines, 27 conics, and 9 corners identified; 68 redundant groups subsumed; 48 total simple replacement tokens generated, 52 tokens propagated. Iteration 2: 102 links created; 43 straight lines, 16 conics, 9 corners, 3 cusps identified; 22 redundant groups subsumed; 37 total simple replacement tokens generated, 58 tokens propagated. Iteration 3: 98 links created; 28 straight lines, 8 conics, 34 corners, and 4 cusps identified; 12 redundant groups subsumed; 24 total simple replacement tokens generated, 62 tokens propagated. Iteration 4: 48 links created; 10 straight lines, 16 conics, 28 corners, and 15 cusps identified; 9 redundant groups subsumed; 19 total simple replacement tokens generated, 44 tokens propagated.

Big M. A (32 × 32) detail of an original (128 × 128) image of a calligraphic "m", this example shows system performance on a nearly binary image. The image presents extended stuctures composed of simple monotone curves and straight lines. Thus, simple replacements alone should suffice in most instances. Figure 7a is the original (128 × 128) gray scale image; and it marks the (64 × 64) area from which the (32 × 32) sub-image is averaged; Figure 7b shows the initial edge tokens overlayed on the attenuated intensity data of this (32 × 32) sub-image. Because the image is nearly binary and because no filtering is used, many spurious edges appear—due to the wanderings of the zero-crossing contours in essentially flat areas. These might have been eliminated by thresholding the edge-filter response, but they serve to contrast the density of grouping along coherent structure with that in ill-defined, in this case, flat areas. Figures 7c and 7d show the output of the first replacement cycle as respectively all active tokens and new tokens only. Figures 7e and 7f present the same information for the second replacement

[10] The absence of corner replacements also contributes to the confusion, as corners would provide anchor points for later grouping

cycle. Throughout, type-two redundancies are apparent as a dense bands of feathered conics and straight lines all along the perimeter of the "m" and the bounding circular arc. These are particularly apparent in 7d and 7f, where only new tokens are displayed. Such redundancies are also observable along the horizontal line in the top region of the image. Nonetheless, it is important to note that these bands are densely arranged about the underlying curves, so that collateral grouping promises an effective strategy for assimilating type-two redundancies into more unified structures.

Statistics for the first two iterations are as follows. Iteration 1: 734 links created; 366 straight lines, 148 conics, 25 corners, and 30 cusps identified; 184 redundant groups subsumed; 332 total simple replacement tokens generated, 250 tokens propagated. Iteration 2: 2564 links created; 220 straight lines, 306 conics, 82 corners, and 16 cusps identified; 103 redundant groups subsumed; 435 total simple replacement tokens generated, 286 tokens propagated.

4 Conclusions

Perceptual grouping on symbolic tokens provides a mechanism for discovering structure underlying seemingly disparate data. However, any grouping is inherently combinatoric. By determining a set of models of significant structure and by operating in an iterative/hierarchical fashion it is possible to manage computational complexity. Local pairwise geometric constraints (imposed by the models) are exploited to build a reduced search graph of tokens (linking). Neighborhoods of this graph are then analyzed for instances of the models (grouping). However, because the search for significant structure is conducted in a neighborhood centered at each token, redundancies of description can and do arise. These show up as feathered patterns of curves and lines in the results and have an explosive effect on graph complexity. Certain redundancies (type-one) are effectively eliminated by detecting sub-sequences. Others (type-two) require a more sophisticated strategy entailing collateral grouping. Finally, and most importantly, the search graph is simplified by encapsulating multiple tokens as single tokens (replacement). With each iteration a less-local graph is constructed and searched. The result is a hierarchical description of significant structure.

Our short-term goals include: making the complex replacements operational and eliminating redundancies of description via a restricted form of collateral grouping. Long term goals include: treating junctions and intersections, explicitly recognizing closed contours, and incorporating full collateral grouping—i.e., grouping over 2D areas instead of just along curves.

Acknowledgements. The authors wish to thank Al Hanson and Ed Riseman for their support and many helpful *suggestions on the manuscript. Special thanks is due* Les Kitchen, now at University of Western Australia, for his encouragement and develop many of the ideas contained herein.

References

[1] Albano, A., "Representation of Digitized Contours in Terms of Conic Arcs and Straight-Line Segments," *Computer Graphics and Image Processing*, Vol 3, pp. 23–33, 1974.

[2] Asada, H. and M. Brady, "The Curvature Primal Sketch," MIT AI-Memo 758, 1984.

[3] Boldt, M. and R. Weiss, "Token-Based Extraction of Straight Lines," UMASS, COINS Tech. Report 87-104, 1987.

[4] Canny, J., "A Computational Approach to Edge Detection", *IEEE PAMI*, Vol 8, No 6, pp. 679–698, 1986.

[5] Davis, L., "A Survey of Edge Detection Techniques", *Computer Graphics and Image Processing*, Vol 4, pp. 248–270, 1975.

[6] de Boor, C., *A Practical Guide to Splines*, Springer-Verlag, 1978.

[7] Dolan, J., "A System for the Perceptual Organization of Image Curves Based on Co-curving Behavior," UMASS, COINS Tech. Report, forthcoming.

[8] Fischler, M. and R. Bolles, "Perceptual Organization and the Curve Partitioning Problem," *IEEE PAMI*, Vol. 8, No. 1, pp. 100–105, 1986.

[9] Haralick, R.M., "Digital Step Edges from Zero Crossings of Second Directional Derivatives", *IEEE PAMI*, Vol 6, pp. 58–68, 1984.

[10] Hochberg, J.E. , "Effects of the Gestalt Revolution: The Cornell Symposium on perception," *Psychological Review*, 64, 2, pp. 73–84, 1957.

[11] Hoffman, D. and W. Richards, "Parts of Recognition," *Cognition*, 18, pp. 65-96, 1985.

[12] Kanizsa, G., *Organization in Vision: Essays on Gestalt Psychology*, Praeger, New York, 1979.

[13] Lowe, D.G. and T.O. Binford, "Segmentation and Aggregation: An Approach to Figure Ground Phenomena," DARPA Image Understanding Workshop, 1982, pp. 168–178.

[14] Lowe, D.G., *Perceptual Organization and Visual Recognition*, Kluwer Academic Publishers, 1985.

[15] Marimont, D., "A Representation for Image Curves," Proce•dings of AAAI, pp. 237-242, 1984.

[16] Marr, D., *Vision*, Freeman, San Francisco, 1982.

[17] Mokhtarian, F. and A. Mackworth, "Scale-based Description and Recognition of Planar Curves," *IEEE PAMI*, 8(1), pp. 34-43, 1986.

[18] Nalwa, V., and E. Pauchon, "Edgel-Aggregation and Edge-Description," DARPA Image Understanding Workshop, 1985, pp. 176-185.

[19] Parent, P. and S. Zucker, "Trace Inference, Curvature Consistency, and Curve Detection," McGill Univ. Tech. Report CIM-86-3, June 1985.

[20] Plass, M. and M. Stone, "Curve-Fitting with Piecewise Parametric Cubics," *Computer Graphics*, Vol. 17, No. 3, pp. 229-239, July 1983.

[21] Pavlidis, T., "Curve Fitting with Conic Splines," *ACM Transactions on Graphics*, Vol. 2, No. 1, pp. 1-31, January 1983.

[22] Rosenfeld, A., "Pyramid Algorithms for Perceptual Organization," *Behavior, Research, Methods, Instruments, & Computers*, 18 (6), pp. 595-600, 1986.

[23] Stevens, K. and A. Brookes, "Detecting Structure by Symbolic Constructions on Tokens," *CVGIP* 37, pp. 238-260, 1987.

[24] Witkin, A., "Scale-Space Filtering," Proceedings of IJCAI, pp. 1019 1021, Karlsruhe, 1983.

[25] Witkin, A. and J. Tenenbaum, "On the Role of Structure in Vision," *Human and Machine Vision*, J. Beck, B. Hope, and A. Rosenfeld, eds., Academic Press, pp. 481-543, 1983.

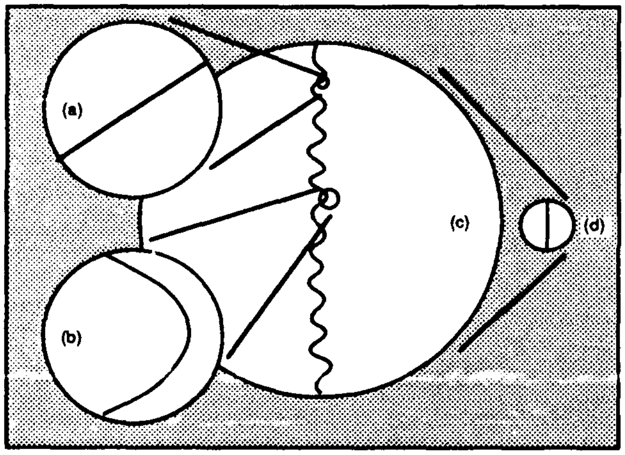


Figure 1. Role of scale in perceived structure. At a very fine scale(a), the structure appears straight, but at a slightly larger scale (b), it appears as a simple curve. At a still larger scale (c), complex curving behavior is apparent. Finally, at a very large scale(d), the strucutre again appears straight.

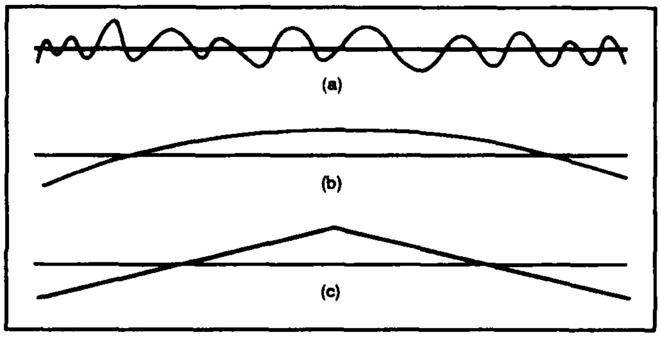


Figure 3. The effect of structural information on the appropriateness of description. Each structure (a), (b), and (c) has roughly the same fit-error with respect a best-fit line. However, the straight-line descriptor is perceptually appropriate in (a); less so in (b); and undesirable in (c).

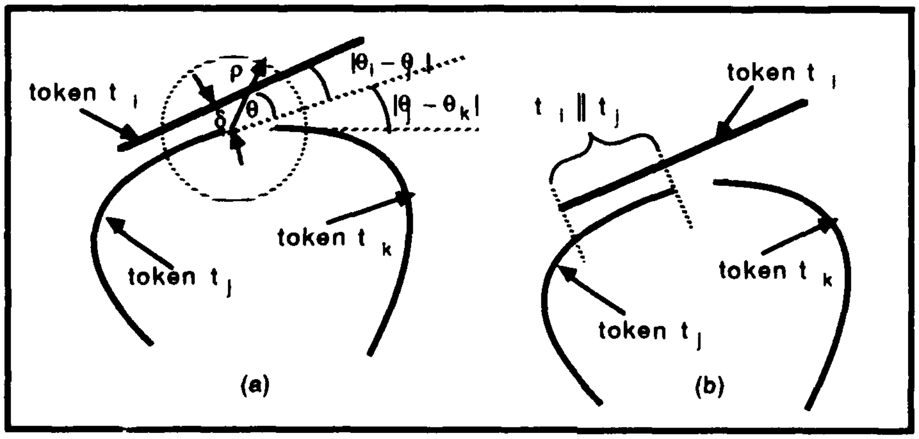


Figure 2. Link criteria. Link angle and link radius are shown in (a): θ = maximum link angle; ρ = maximum link radius; δ = distance and $|\theta_i - \theta_j|$ = angular difference between tokens t_i and t_j. Link overlap is shown in (b): $t_i \| t_j$ is the overlap of token t_i onto token t_j.

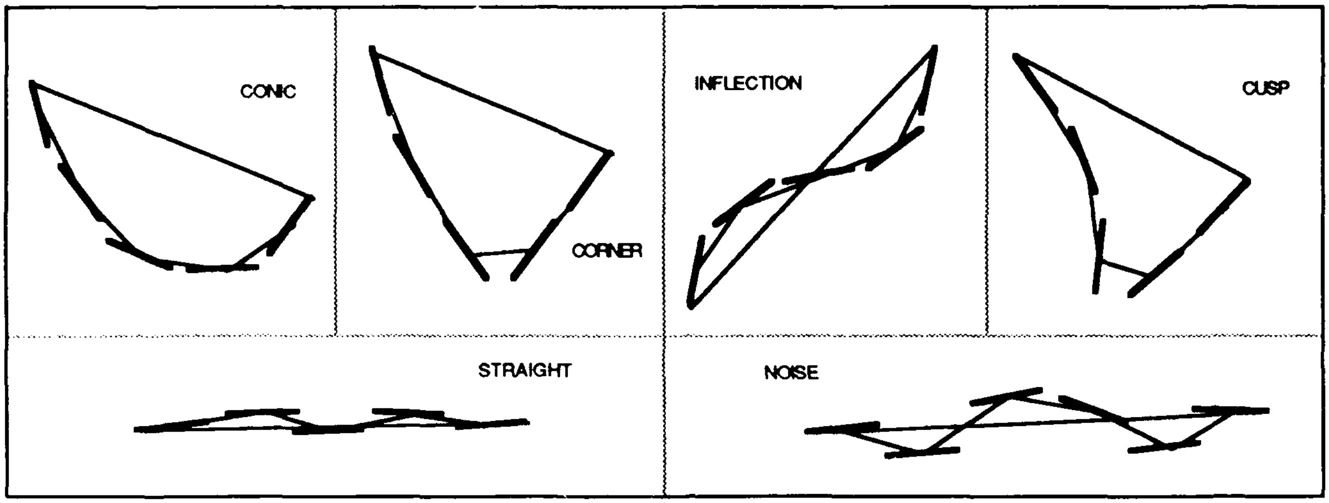


Figure 4. Simple examples of classes of significant curvilinear structures. Polygonal approximations (thin lines) are superimposed on intial unit tangent tokens (thick lines).

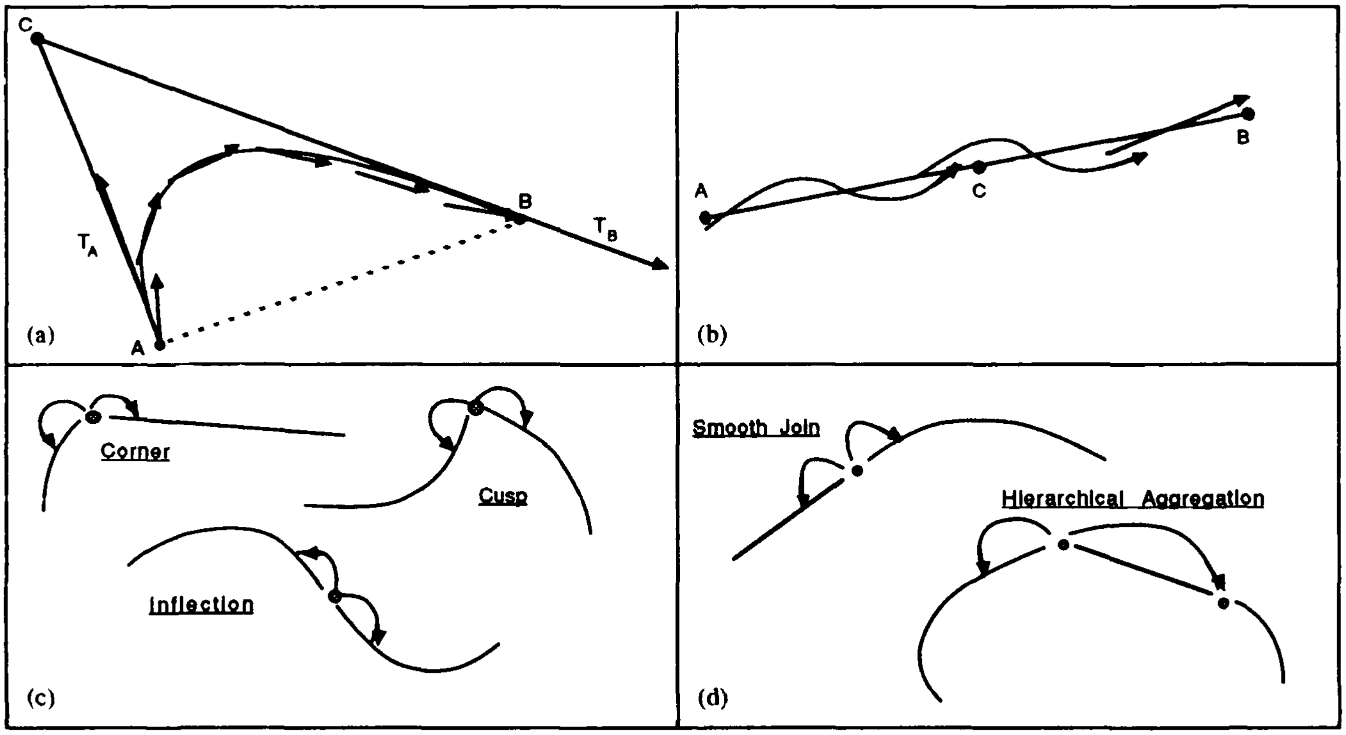


Figure 5. Types of replacement: Simple Replacements in Figures 5(a) and 5(b); Complex Replacements in Figures 5(c) and 5(d). In 5(a), a sequence of linear tokens is replaced by a single conic-spline token. The guiding polygon ACB and the endpoint tangents T_A and T_B of the new spline are also shown. In Figure 5(b), a sequence of inflected curve tokens and a straight token are replaced by a single straight-line token. In Figure 5(c), examples are shown of point-event tokens: inflection, corner, and cusp. These effect a concatenation of simple descriptors yielding a single replacement token. This new descriptor is usually a triple of the form:(simple descriptor, point token, simple descriptor). Figure 5(d) shows another useful point token, the smooth-join which is not directly related to the set of significant curve structure but allows closing of forms. Also shown in Figure 5(d), point-event tokens can be composed in hieracrchical fashion to yield arbitrarily complex descriptions.

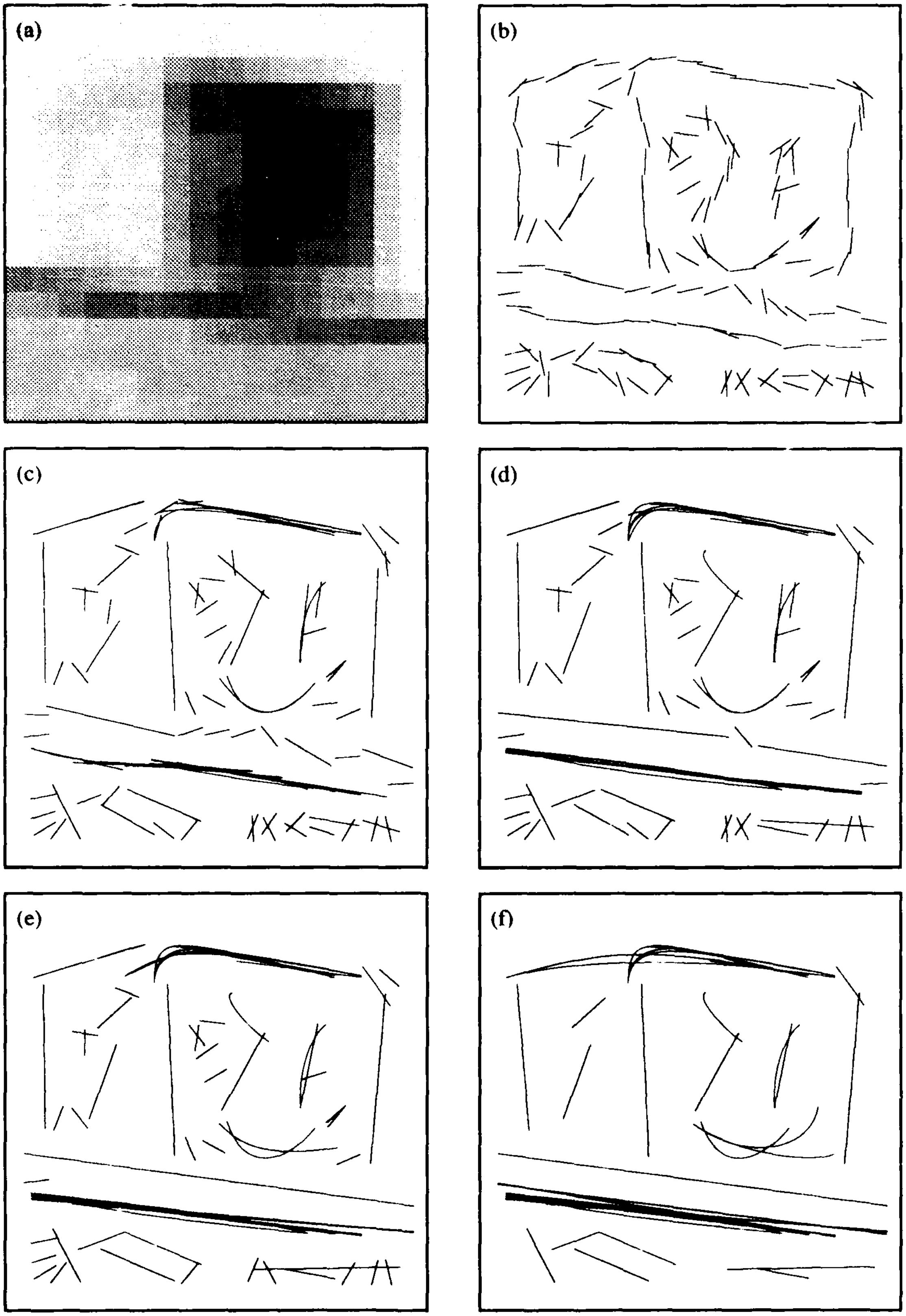


Figure 6. Intensity image (16 x 16) detail 6(a). Initial 150 unit tangent tokens 6(b). Results of 1st—4th iterations 6(c)—6(f). Relevant statistics (lr = link radius, pr = perceptual radius, nt = new tokens): (c) lr = 0.25, pr = 2.5, nt = 48; (d) lr = 0.375, pr = 3.75, nt = 37; (e) lr = 0.5625, pr = 5.625, nt = 24; (f) lr = 0.84375, pr = 8.4375, nt = 19.

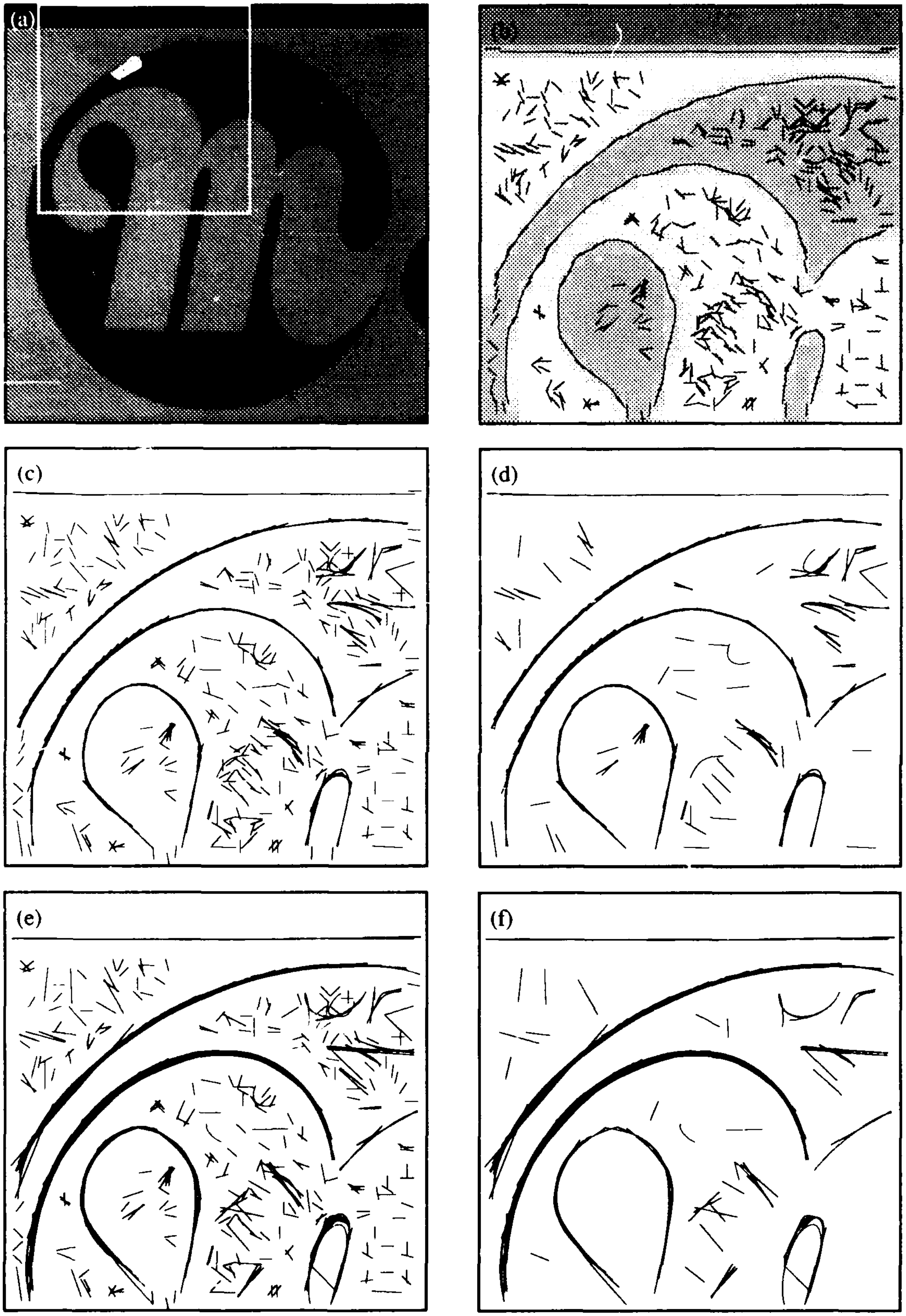
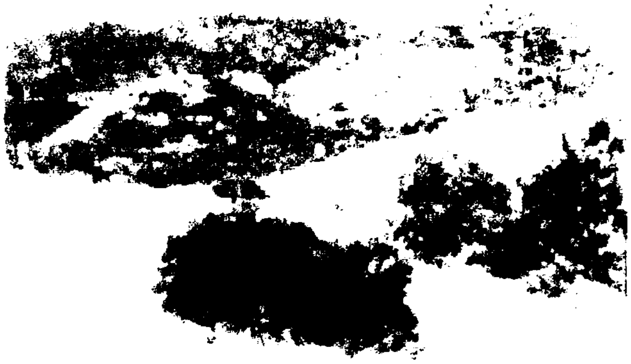
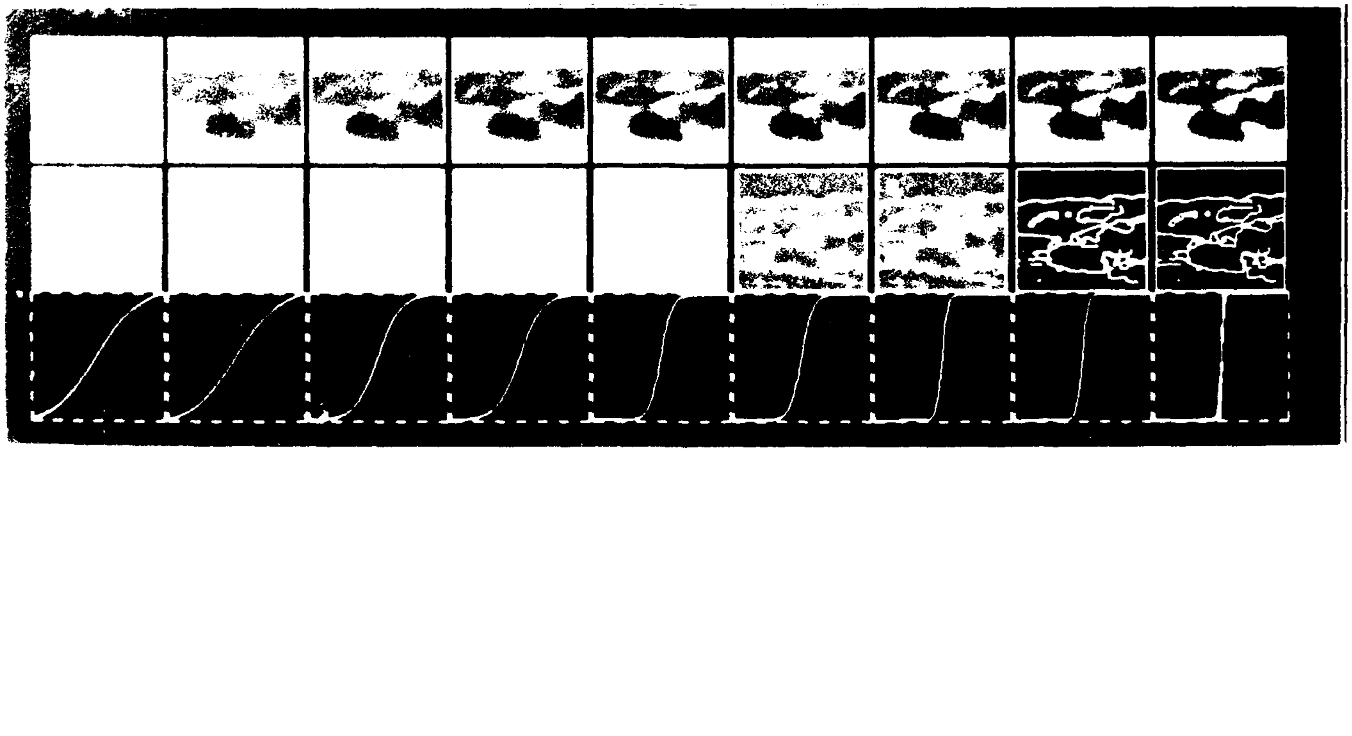


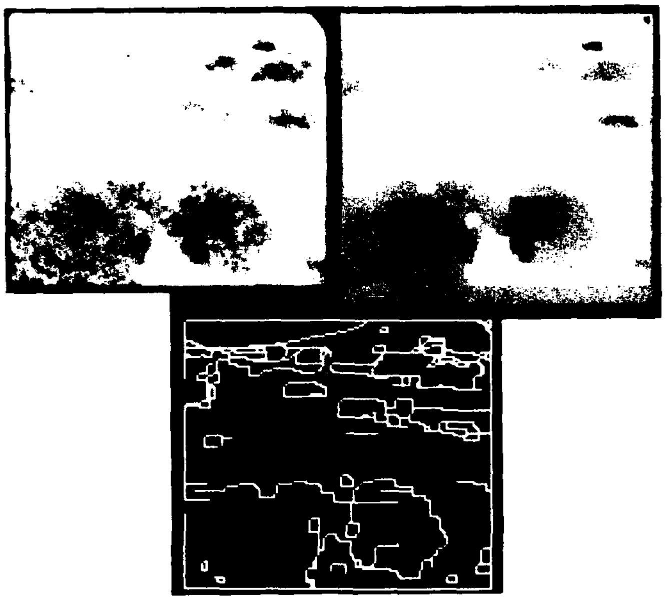
Figure 7. Original (128 x 128) intensity image, with subimage area indicated 7(a). Initial 629 *unit tangent tokens* overlayed on attenuated (32 x 32) intensity data 7(b). Results of 1st iteration: 7(c) all tokens, 7(d) new tokens only. Results of 2nd iteration: 7(e) all tokens, 7(f) new tokens only. Relevant statistics (lr = link radius, pr = perceptual radius, nt = new tokens): (c,d) lr = 0.25, pr = 2.5, nt = 296; (e,f) lr = 0.375, pr = 3.75, nt = 383.

REFERENCE

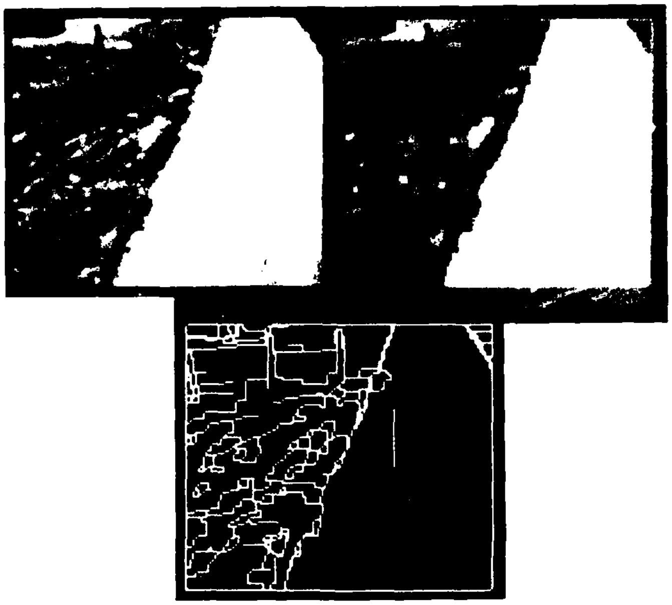
Three (3) Color Photographs from the paper, "Color
Image Segmentation Using Markov Random Fields",
Michael J. Daily; Hughes Research Laboratories........ 1147





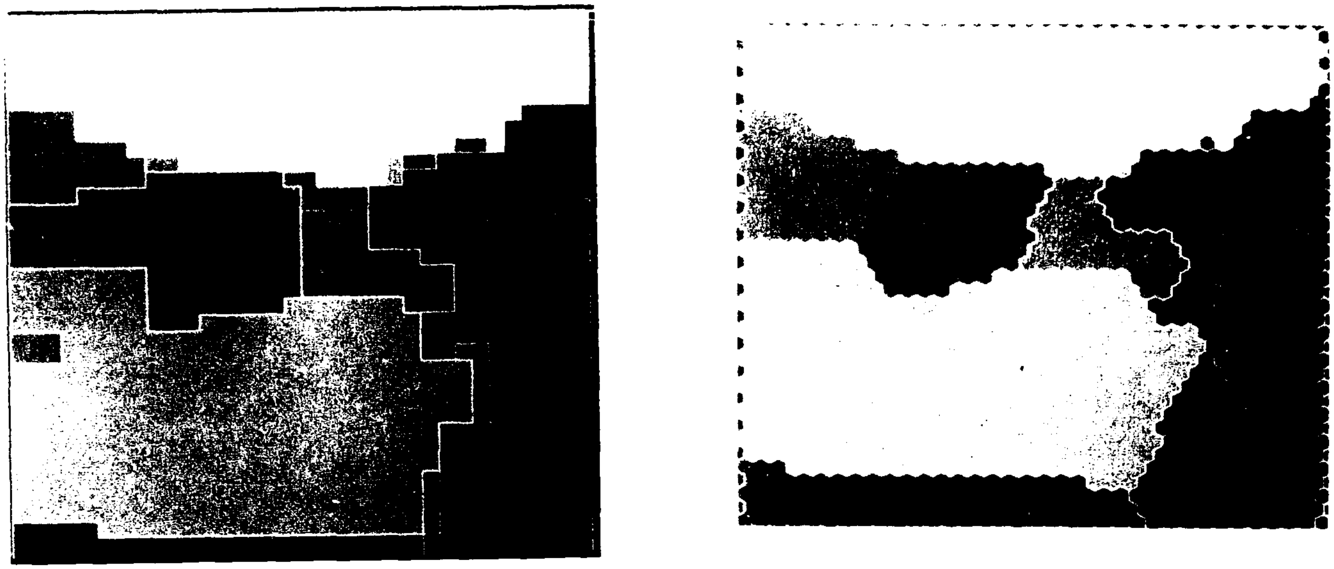


Color Photo 4. MRF segmentation on 128x128 rectangular lattice using Euclidean RGB color difference metric. (Top left) original image, (top right) final segmented image, (bottom) final line process discontinuities



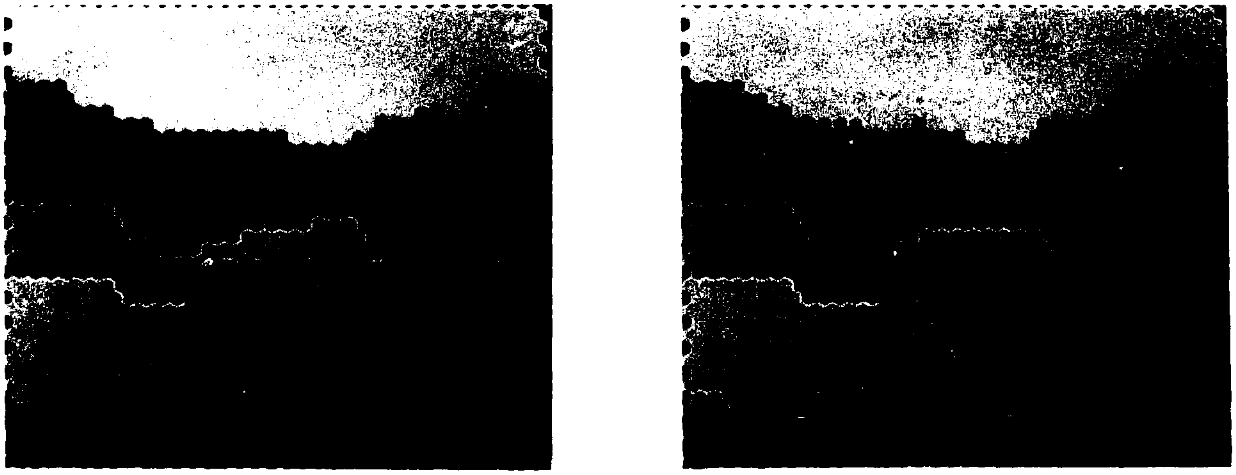
Color Photo 5. MRF segmentation on 128x128 rectangular lattice using Euclidean RGB color difference metric. (Top left) original image, (top right) final segmented image, (bottom) final line process discontinuities.

Color Photo 6. Hexagonally sampled image from rectangular image of Figure 1A using hexagons of width 8 pixels (creates a 64x64 hexagonal lattice).



Color Photo 7. Comparison of rectangular and hexagonal MRFs using a 35x40 lattice (14 pixel width hexagons). The rectangular MRF on the left displays a marked bias toward unnatural right angles.

Color Photo 8. Segmentation results for Color Photo 2 using a 64x64 hexagonal lattice and the Euclidean RGB color difference metric.



Color Photo 9. A comparison of two color difference methods computed using the same parameter values. Left, Euclidean RGB segmentation. Right, Maximum IHS segmentation. The IHS method produces excessive fragmentation since hue is a more sensitive difference measure at equal line process penalty values than RGB.

END
FILMED
11-89
DTIC